

PHP 4.0 Manuel de Référence.

Table of Contents

1	Preface [Notes en ligne]	1
1.1	Les auteurs [Notes en ligne]	1
1.2	Copyright [Notes en ligne]	1
2	Préface [Notes en ligne]	2
2.1	A propos de ce manuel [Notes en ligne]	2
3	Copyright, distribution, historique [Notes en ligne]	3
4	Installation [Notes en ligne]	4
4.1	Télécharger la dernière version [Notes en ligne]	4
4.2	Installation sous UNIX [Notes en ligne]	4
4.2.1	Installation rapide (Version Module Apache) [Notes en ligne]	4
4.2.2	Module fhttpd [Notes en ligne]	6
4.2.3	Autres serveurs web [Notes en ligne]	6
4.2.4	`/usr/local/src/fhttpd' [Notes en ligne]	6
4.2.5	Options de base de données [Notes en ligne]	6
4.2.6	Compilation [Notes en ligne]	6
4.2.7	Tests [Notes en ligne]	7
4.2.8	Performances [Notes en ligne]	7
4.3	Liste complète des options de configuration [Notes en ligne]	7
4.3.1	Configuration pour le support des bases de données [Notes en ligne]	7
4.3.1.1	install.configure.with-adabas [Notes en ligne]	8
4.3.1.2	install.configure.enable-dba [Notes en ligne]	8
4.3.1.3	install.configure.enable-dbase [Notes en ligne]	8
4.3.1.4	install.configure.with-dbase [Notes en ligne]	8
4.3.1.5	install.configure.with-db2 [Notes en ligne]	8
4.3.1.6	install.configure.with-db3 [Notes en ligne]	8
4.3.1.7	install.configure.with-dbm [Notes en ligne]	9
4.3.1.8	install.configure.with-dbmaker [Notes en ligne]	9
4.3.1.9	install.configure.with-empres [Notes en ligne]	9
4.3.1.10	install.configure.enable-filepro [Notes en ligne]	9
4.3.1.11	install.configure.with-filepro [Notes en ligne]	9
4.3.1.12	install.configure.with-gdbm [Notes en ligne]	9
4.3.1.13	install.configure.with-hyperwave [Notes en ligne]	9
4.3.1.14	install.configure.with-ibm-db2 [Notes en ligne]	10
4.3.1.15	install.configure.with-informix [Notes en ligne]	10
4.3.1.16	install.configure.with-ingres [Notes en ligne]	10
4.3.1.17	install.configure.with-interbase [Notes en ligne]	10
4.3.1.18	install.configure.with-ldap [Notes en ligne]	10
4.3.1.19	install.configure.with-mysql [Notes en ligne]	11
4.3.1.20	install.configure.with-mysql [Notes en ligne]	11
4.3.1.21	install.configure.with-ndbm [Notes en ligne]	11
4.3.1.22	install.configure.with-oci8 [Notes en ligne]	11
4.3.1.23	install.configure.with-oracle [Notes en ligne]	11
4.3.1.24	install.configure.with-pgsql [Notes en ligne]	12
4.3.1.25	install.configure.with-solid [Notes en ligne]	12

Table of Contents

4.3.1.26	install.configure.with-sybase-ct [Notes en ligne]	12
4.3.1.27	install.configure.with-sybase [Notes en ligne]	12
4.3.1.28	install.configure.with-openlink [Notes en ligne]	13
4.3.1.29	install.configure.with-iodbc [Notes en ligne]	13
4.3.1.30	install.configure.with-custom-odbc [Notes en ligne]	13
4.3.1.31	install.configure.disable-unified-odbc [Notes en ligne]	13
4.3.1.32	install.configure.with-unixODBC [Notes en ligne]	14
4.3.1.33	install.configure.with-velocis [Notes en ligne]	14
4.3.2	Ecommerce [Notes en ligne]	14
4.3.2.1	install.configure.with-ccvs [Notes en ligne]	14
4.3.2.2	install.configure.with-mck [Notes en ligne]	14
4.3.2.3	install.configure.with-cybercash [Notes en ligne]	14
4.3.2.4	install.configure.with-pfpro [Notes en ligne]	15
4.3.3	Graphisme [Notes en ligne]	15
4.3.3.1	install.configure.enable-freetype-4bit-antialias-hack [Notes en ligne]	15
4.3.3.2	install.configure.with-gd [Notes en ligne]	15
4.3.3.3	install.configure.without-gd [Notes en ligne]	15
4.3.3.4	install.configure.with-imagick [Notes en ligne]	15
4.3.3.5	install.configure.with-jpeg-dir [Notes en ligne]	15
4.3.3.6	install.configure.with-png-dir [Notes en ligne]	16
4.3.3.7	install.configure.enable-t1lib [Notes en ligne]	16
4.3.3.8	install.configure.with-t1lib [Notes en ligne]	16
4.3.3.9	install.configure.with-tiff-dir [Notes en ligne]	16
4.3.3.10	install.configure.with-ttf [Notes en ligne]	16
4.3.3.11	install.configure.with-xpm-dir [Notes en ligne]	16
4.3.4	Divers [Notes en ligne]	17
4.3.4.1	install.configure.disable-bcmath [Notes en ligne]	17
4.3.4.2	install.configure.disable-display-source [Notes en ligne]	17
4.3.4.3	install.configure.disable-libtool-lock [Notes en ligne]	17
4.3.4.4	install.configure.disable-pear [Notes en ligne]	17
4.3.4.5	install.configure.disable-pic [Notes en ligne]	17
4.3.4.6	install.configure.disable-posix [Notes en ligne]	18
4.3.4.7	install.configure.disable-rpath [Notes en ligne]	18
4.3.4.8	install.configure.disable-session [Notes en ligne]	18
4.3.4.9	install.configure.enable-bcmath [Notes en ligne]	18
4.3.4.10	install.configure.enable-c9x-inline [Notes en ligne]	18
4.3.4.11	install.configure.enable-calendar [Notes en ligne]	18
4.3.4.12	install.configure.enable-debug [Notes en ligne]	19
4.3.4.13	install.configure.enable-debugger [Notes en ligne]	19
4.3.4.14	install.configure.enable-discard-path [Notes en ligne]	19
4.3.4.15	install.configure.enable-dmalloc [Notes en ligne]	19
4.3.4.16	install.configure.enable-exif [Notes en ligne]	19
4.3.4.17	install.configure.enable-experimental-zts [Notes en ligne]	19
4.3.4.18	install.configure.enable-fast-install [Notes en ligne]	19
4.3.4.19	install.configure.enable-force-cgi-redirect [Notes en ligne]	20
4.3.4.20	install.configure.enable-inline-optimization [Notes en ligne]	20
4.3.4.21	install.configure.enable-libgcc [Notes en ligne]	20

Table of Contents

4.3.4.22	install.configure.enable-maintainer-mode [Notes en ligne]	20
4.3.4.23	install.configure.enable-memory-limit [Notes en ligne]	20
4.3.4.24	install.configure.enable-safe-mode [Notes en ligne]	20
4.3.4.25	install.configure.enable-satellite [Notes en ligne]	21
4.3.4.26	install.configure.enable-shared [Notes en ligne]	21
4.3.4.27	install.configure.enable-sigchild [Notes en ligne]	21
4.3.4.28	install.configure.enable-static [Notes en ligne]	21
4.3.4.29	install.configure.enable-sysvsem [Notes en ligne]	21
4.3.4.30	install.configure.enable-sysvshm [Notes en ligne]	21
4.3.4.31	install.configure.enable-trans-sid [Notes en ligne]	21
4.3.4.32	install.configure.with-cdb [Notes en ligne]	22
4.3.4.33	install.configure.with-config-file-path [Notes en ligne]	22
4.3.4.34	install.configure.with-cpdflib [Notes en ligne]	22
4.3.4.35	install.configure.with-esoob [Notes en ligne]	22
4.3.4.36	install.configure.with-exec-dir [Notes en ligne]	22
4.3.4.37	install.configure.with-fdftk [Notes en ligne]	22
4.3.4.38	install.configure.with-gnu-ld [Notes en ligne]	23
4.3.4.39	install.configure.with-icap [Notes en ligne]	23
4.3.4.40	install.configure.with-imap [Notes en ligne]	23
4.3.4.41	install.configure.with-imsp [Notes en ligne]	23
4.3.4.42	install.configure.with-java [Notes en ligne]	23
4.3.4.43	install.configure.with-kerberos [Notes en ligne]	23
4.3.4.44	install.configure.with-mcal [Notes en ligne]	24
4.3.4.45	install.configure.with-mcrypt [Notes en ligne]	24
4.3.4.46	install.configure.with-mhash [Notes en ligne]	24
4.3.4.47	install.configure.with-mm [Notes en ligne]	24
4.3.4.48	install.configure.with-mod_charset [Notes en ligne]	24
4.3.4.49	install.configure.with-pdflib [Notes en ligne]	24
4.3.4.50	install.configure.with-readline [Notes en ligne]	24
4.3.4.51	install.configure.with-regex [Notes en ligne]	25
4.3.4.52	install.configure.with-servlet [Notes en ligne]	25
4.3.4.53	install.configure.with-swf [Notes en ligne]	25
4.3.4.54	install.configure.with-system-regex [Notes en ligne]	25
4.3.4.55	install.configure.with-tsrml-pth [Notes en ligne]	25
4.3.4.56	install.configure.with-tsrml-threads [Notes en ligne]	25
4.3.4.57	install.configure.with-x [Notes en ligne]	26
4.3.4.58	install.configure.with-zlib-dir [Notes en ligne]	26
4.3.4.59	install.configure.with-zlib [Notes en ligne]	26
4.3.4.60	install.configure.without-pcre-regex [Notes en ligne]	26
4.3.4.61	install.configure.without-posix [Notes en ligne]	26
4.3.5	Réseau [Notes en ligne]	26
4.3.5.1	install.configure.with-curl [Notes en ligne]	26
4.3.5.2	install.configure.enable-ftp [Notes en ligne]	27
4.3.5.3	install.configure.with-ftp [Notes en ligne]	27
4.3.5.4	install.configure.disable-url-fopen-wrapper [Notes en ligne]	27
4.3.5.5	install.configure.with-mod-dav [Notes en ligne]	27
4.3.5.6	install.configure.with-openssl [Notes en ligne]	27

Table of Contents

4.3.5.7	install.configure.with-snmp [Notes en ligne]	27
4.3.5.8	install.configure.enable-ucd-snmp-hack [Notes en ligne]	28
4.3.5.9	install.configure.enable-sockets [Notes en ligne]	28
4.3.5.10	install.configure.with-yaz [Notes en ligne]	28
4.3.5.11	install.configure.enable-yp [Notes en ligne]	28
4.3.5.12	install.configure.with-yp [Notes en ligne]	28
4.3.6	Configuration de PHP [Notes en ligne]	28
4.3.6.1	install.configure.enable-magic-quotes [Notes en ligne]	28
4.3.6.2	install.configure.disable-short-tags [Notes en ligne]	29
4.3.6.3	install.configure.enable-track-vars [Notes en ligne]	29
4.3.7	Serveur [Notes en ligne]	29
4.3.7.1	install.configure.with-aolserver-src [Notes en ligne]	29
4.3.7.2	install.configure.with-aolserver [Notes en ligne]	29
4.3.7.3	install.configure.with-apache [Notes en ligne]	29
4.3.7.4	install.configure.with-apxs [Notes en ligne]	29
4.3.7.5	install.configure.enable-versioning [Notes en ligne]	30
4.3.7.6	install.configure.with-fhttpd [Notes en ligne]	30
4.3.7.7	install.configure.with-nsapi [Notes en ligne]	30
4.3.7.8	install.configure.with-phhttpd [Notes en ligne]	30
4.3.7.9	install.configure.with-pi3web [Notes en ligne]	30
4.3.7.10	install.configure.with-roxen [Notes en ligne]	30
4.3.7.11	install.configure.enable-roxen-zts [Notes en ligne]	31
4.3.7.12	install.configure.with-thhttpd [Notes en ligne]	31
4.3.7.13	install.configure.with-zeus [Notes en ligne]	31
4.3.8	Texte et langue [Notes en ligne]	31
4.3.8.1	install.configure.with-aspell [Notes en ligne]	31
4.3.8.2	install.configure.with-gettext [Notes en ligne]	31
4.3.8.3	install.configure.with-pspell [Notes en ligne]	31
4.3.8.4	install.configure.with-recode [Notes en ligne]	32
4.3.9	XML [Notes en ligne]	32
4.3.9.1	install.configure.with-dom [Notes en ligne]	32
4.3.9.2	install.configure.enable-sablot-errors-descriptive [Notes en ligne]	32
4.3.9.3	install.configure.with-sablot [Notes en ligne]	32
4.3.9.4	install.configure.enable-wddx [Notes en ligne]	32
4.3.9.5	install.configure.disable-xml [Notes en ligne]	32
4.3.9.6	install.configure.with-xml [Notes en ligne]	33
4.4	Installation sous Windows 95/98/NT [Notes en ligne]	33
4.4.1	Installation [Notes en ligne]	33
4.4.2	Windows 95/98/NT et PWS/IIS 3 [Notes en ligne]	34
4.4.3	Windows NT et IIS 4 [Notes en ligne]	35
4.4.4	Windows 9x/NT et Apache 1.3.x [Notes en ligne]	35
4.4.5	Omni HTTPd 2.0b1 pour Windows [Notes en ligne]	35
4.4.6	Installshield [Notes en ligne]	36
4.4.7	Modules PHP [Notes en ligne]	36
4.5	Problèmes? [Notes en ligne]	37
4.5.1	Lisez la FAQ [Notes en ligne]	37
4.5.2	Rapport de bug [Notes en ligne]	37

Table of Contents

4.5.3	Autres problèmes [Notes en ligne]	37
5	Introduction [Notes en ligne]	38
5.1	Qu'est ce que PHP? [Notes en ligne]	38
5.2	Que peut faire PHP? [Notes en ligne]	38
5.3	La génèse du PHP [Notes en ligne]	39
6	Sécurité [Notes en ligne]	41
6.1	Binaires CGI [Notes en ligne]	41
6.1.1	Faiblesses connues [Notes en ligne]	41
6.1.2	Cas 1: Tous les fichiers sont publics [Notes en ligne]	42
6.1.3	Cas 2: Utilisation de la directive de compilation <code>--enable-force-cgi-redirect</code> [Notes en ligne]	42
6.1.4	Cas 3: Utilisation du "doc_root" ou du "user_dir" [Notes en ligne]	42
6.1.5	Cas 4: L'exécutable PHP à l'extérieur de l'arborescence du serveur [Notes en ligne]	43
6.2	Module Apache [Notes en ligne]	43
6.3	Sécurité des fichiers [Notes en ligne]	44
6.4	Rapport d'erreur [Notes en ligne]	45
6.5	User Submitted Data [Notes en ligne]	46
6.6	Considérations générales [Notes en ligne]	46
6.7	Etre à jour [Notes en ligne]	47
7	Configuration [Notes en ligne]	48
7.1	Le fichier de configuration [Notes en ligne]	48
7.1.1	Directives de configuration générale [Notes en ligne]	48
7.1.1.1	ini.allow-url-fopen [Notes en ligne]	48
7.1.1.2	ini.asp-tags [Notes en ligne]	49
7.1.1.3	ini.auto-append-file [Notes en ligne]	49
7.1.1.4	ini.auto-prepend-file [Notes en ligne]	49
7.1.1.5	ini.cgi-ext [Notes en ligne]	49
7.1.1.6	ini.display-errors [Notes en ligne]	49
7.1.1.7	ini.doc-root [Notes en ligne]	49
7.1.1.8	ini.engine [Notes en ligne]	50
7.1.1.9	ini.error-log [Notes en ligne]	50
7.1.1.10	ini.error-reporting [Notes en ligne]	50
7.1.1.11	ini.open-basedir [Notes en ligne]	50
7.1.1.12	ini.gpc-order [Notes en ligne]	51
7.1.1.13	ini.ignore-user-abort [Notes en ligne]	51
7.1.1.14	ini.include-path [Notes en ligne]	51
7.1.1.15	ini.isapi-ext [Notes en ligne]	51
7.1.1.16	ini.log-errors [Notes en ligne]	51
7.1.1.17	ini.magic-quotes-gpc [Notes en ligne]	52
7.1.1.18	ini.magic-quotes-runtime [Notes en ligne]	52
7.1.1.19	ini.magic-quotes-sybase [Notes en ligne]	52
7.1.1.20	ini.max-execution-time [Notes en ligne]	52
7.1.1.21	ini.memory-limit [Notes en ligne]	52
7.1.1.22	ini.nsapi-ext [Notes en ligne]	52
7.1.1.23	ini.register-globals [Notes en ligne]	53

Table of Contents

7.1.1.24	ini.short-open-tag [Notes en ligne]	53
7.1.1.25	ini.sql.safe-mode [Notes en ligne]	53
7.1.1.26	ini.track-errors [Notes en ligne]	53
7.1.1.27	ini.track-vars [Notes en ligne]	53
7.1.1.28	ini.upload-tmp-dir [Notes en ligne]	53
7.1.1.29	ini.user-dir [Notes en ligne]	54
7.1.1.30	ini.warn-plus-overloading [Notes en ligne]	54
7.1.2	Configuration des directives concernant le mail [Notes en ligne]	54
7.1.2.1	ini.smtp [Notes en ligne]	54
7.1.2.2	ini.sendmail-from [Notes en ligne]	54
7.1.2.3	ini.sendmail-path [Notes en ligne]	54
7.1.3	Directives de configuration du "Safe Mode" [Notes en ligne]	55
7.1.3.1	ini.safe-mode [Notes en ligne]	55
7.1.3.2	ini.safe-mode-exec-dir [Notes en ligne]	55
7.1.4	Directives de configuration de débbugage. [Notes en ligne]	55
7.1.4.1	ini.debugger.host [Notes en ligne]	55
7.1.4.2	ini.debugger.port [Notes en ligne]	55
7.1.4.3	ini.debugger.enabled [Notes en ligne]	55
7.1.5	Directives de chargement des extensions [Notes en ligne]	55
7.1.5.1	ini.enable-dl [Notes en ligne]	56
7.1.5.2	ini.extension-dir [Notes en ligne]	56
7.1.5.3	ini.extension [Notes en ligne]	56
7.1.6	MySQL Configuration Directives [Notes en ligne]	56
7.1.6.1	ini.mysql.allow-persistent [Notes en ligne]	56
7.1.6.2	ini.mysql.default-host [Notes en ligne]	56
7.1.6.3	ini.mysql.default-user [Notes en ligne]	56
7.1.6.4	ini.mysql.default-password [Notes en ligne]	57
7.1.6.5	ini.mysql.max-persistent [Notes en ligne]	57
7.1.6.6	ini.mysql.max-links [Notes en ligne]	57
7.1.7	Directives de configuration mSQL [Notes en ligne]	57
7.1.7.1	ini.msql.allow-persistent [Notes en ligne]	57
7.1.7.2	ini.msql.max-persistent [Notes en ligne]	57
7.1.7.3	ini.msql.max-links [Notes en ligne]	57
7.1.8	Directives de configuration Postgres [Notes en ligne]	58
7.1.8.1	ini.pgsql.allow-persistent [Notes en ligne]	58
7.1.8.2	ini.pgsql.max-persistent [Notes en ligne]	58
7.1.8.3	ini.pgsql.max-links [Notes en ligne]	58
7.1.9	Directives de configuration SESAM [Notes en ligne]	58
7.1.9.1	ini.sesam-oml [Notes en ligne]	58
7.1.9.2	ini.sesam-configfile [Notes en ligne]	58
7.1.9.3	ini.sesam-messagecatalog [Notes en ligne]	59
7.1.10	Directives de configuration Sybase [Notes en ligne]	59
7.1.10.1	ini.sybase.allow-persistent [Notes en ligne]	59
7.1.10.2	ini.sybase.max-persistent [Notes en ligne]	59
7.1.10.3	ini.sybase.max-links [Notes en ligne]	59
7.1.11	Sybase-CT Configuration Directives [Notes en ligne]	59
7.1.11.1	ini.sybct.allow-persistent [Notes en ligne]	59

Table of Contents

7.1.11.2	ini.sybct.max-persistent [Notes en ligne]	60
7.1.11.3	ini.sybct.max-links [Notes en ligne]	60
7.1.11.4	ini.sybct.min-server-severity [Notes en ligne]	60
7.1.11.5	ini.sybct.min-client-severity [Notes en ligne]	60
7.1.11.6	ini.sybct.login-timeout [Notes en ligne]	60
7.1.11.7	ini.sybct.timeout [Notes en ligne]	60
7.1.11.8	ini.sybct.hostname [Notes en ligne]	61
7.1.12	Directives de configuration Informix [Notes en ligne]	61
7.1.12.1	ini.ifx.allow-persistent [Notes en ligne]	61
7.1.12.2	ini.ifx.max-persistent [Notes en ligne]	61
7.1.12.3	ini.ifx.max-links [Notes en ligne]	61
7.1.12.4	ini.ifx.default-host [Notes en ligne]	61
7.1.12.5	ini.ifx.default-user [Notes en ligne]	61
7.1.12.6	ini.ifx.default-password [Notes en ligne]	62
7.1.12.7	ini.ifx.blobinfile [Notes en ligne]	62
7.1.12.8	ini.ifx.textasvarchar [Notes en ligne]	62
7.1.12.9	ini.ifx.byteasvarchar [Notes en ligne]	62
7.1.12.10	ini.ifx.charasvarchar [Notes en ligne]	62
7.1.12.11	ini.ifx.nullformat [Notes en ligne]	62
7.1.13	Directives de configuration pour les calculs mathématiques. [Notes en ligne]	63
7.1.13.1	ini.bcmath.scale [Notes en ligne]	63
7.1.14	Directives de configuration du navigateur. [Notes en ligne]	63
7.1.14.1	ini.browscap [Notes en ligne]	63
7.1.15	Directives de configuration du driver ODBC unifié [Notes en ligne]	63
7.1.15.1	ini.uodbc.default-db [Notes en ligne]	63
7.1.15.2	ini.uodbc.default-user [Notes en ligne]	63
7.1.15.3	ini.uodbc.default-pw [Notes en ligne]	63
7.1.15.4	ini.uodbc.allow-persistent [Notes en ligne]	64
7.1.15.5	ini.uodbc.max-persistent [Notes en ligne]	64
7.1.15.6	ini.uodbc.max-links [Notes en ligne]	64
8	Caractéristiques [Notes en ligne]	65
8.1	Gestion des connexions [Notes en ligne]	65
8.2	Cookies [Notes en ligne]	66
8.3	Gestion des erreurs [Notes en ligne]	66
8.4	Gestion des chargements de fichier [Notes en ligne]	69
8.4.1	Chargements de fichiers par méthode POST [Notes en ligne]	69
8.4.2	Erreurs classiques [Notes en ligne]	71
8.4.3	Chargement multiples de fichiers [Notes en ligne]	71
8.4.4	Chargement par méthode PUT [Notes en ligne]	71
8.5	Authentification HTTP avec PHP [Notes en ligne]	72
8.6	Création d'images [Notes en ligne]	74
8.7	Connexions persistantes aux bases de données [Notes en ligne]	74
8.8	Utilisation des fichiers à distance [Notes en ligne]	75
9	Langage [Notes en ligne]	77
9.1	La syntaxe de base [Notes en ligne]	77

Table of Contents

9.1.1 Le passage du HTML au PHP [Notes en ligne]	77
9.1.2 Le séparateur d'instruction [Notes en ligne]	77
9.1.3 Commentaires [Notes en ligne]	78
9.2 Les constantes [Notes en ligne]	78
9.3 Les structures de contrôle [Notes en ligne]	79
9.3.1 if [Notes en ligne]	80
9.3.2 else [Notes en ligne]	80
9.3.3 elseif [Notes en ligne]	81
9.3.4 Syntaxe alternative [Notes en ligne]	81
9.3.5 while [Notes en ligne]	82
9.3.6 do..while [Notes en ligne]	83
9.3.7 for [Notes en ligne]	83
9.3.8 foreach [Notes en ligne]	84
9.3.9 break [Notes en ligne]	86
9.3.10 continue [Notes en ligne]	86
9.3.11 switch [Notes en ligne]	87
9.3.12 require() [Notes en ligne] [Exemples]	89
9.3.13 include() [Notes en ligne] [Exemples]	89
9.3.14 require_once() [Notes en ligne] [Exemples]	92
9.3.15 include_once() [Notes en ligne] [Exemples]	93
9.4 Les expressions [Notes en ligne]	94
9.5 Fonctions [Notes en ligne]	96
9.5.1 Les fonctions utilisateurs [Notes en ligne] [Exemples]	96
9.5.2 Les arguments de fonction [Notes en ligne] [Exemples]	96
9.5.2.1 Passage d'arguments par référence [Notes en ligne] [Exemples]	97
9.5.2.2 Valeur par défaut des arguments [Notes en ligne] [Exemples]	97
9.5.2.3 Nombre d'arguments variable [Notes en ligne] [Exemples]	98
9.5.3 Les valeurs de retour [Notes en ligne] [Exemples]	98
9.5.4 old_function [Notes en ligne] [Exemples]	99
9.5.5 Variable functions [Notes en ligne] [Exemples]	99
9.6 Classes et objets [Notes en ligne]	100
9.6.1 Les classes : class [Notes en ligne]	100
9.7 Les opérateurs [Notes en ligne]	102
9.7.1 Les opérateurs arithmétiques [Notes en ligne]	102
9.7.2 Les opérateurs d'assignement [Notes en ligne]	102
9.7.3 Bitwise Operators [Notes en ligne]	103
9.7.4 Opérateurs de comparaison [Notes en ligne]	103
9.7.5 Opérateur de contrôle d'erreur [Notes en ligne]	104
9.7.6 Opérateur d'exécutions [Notes en ligne]	104
9.7.7 Opérateurs d'incrementation/Décrementation [Notes en ligne]	105
9.7.8 Les opérateurs logiques [Notes en ligne]	105
9.7.9 La précédence des opérateurs [Notes en ligne]	106
9.7.10 Opérateurs de chaînes [Notes en ligne]	106
9.8 Types [Notes en ligne]	107
9.8.1 Entiers [Notes en ligne]	107
9.8.2 Les nombres à virgule flottante [Notes en ligne]	107
9.8.3 Les chaînes de caractères [Notes en ligne]	108

Table of Contents

9.8.3.1	Conversion de type [Notes en ligne]	110
9.8.4	Les tableaux [Notes en ligne]	110
9.8.4.1	Tableaux à une dimension [Notes en ligne]	110
9.8.4.2	Tableaux à plusieurs dimensions [Notes en ligne]	111
9.8.5	Les objets [Notes en ligne]	112
9.8.5.1	Initialisation d'un objet [Notes en ligne]	112
9.8.6	Définition du type [Notes en ligne]	113
9.8.6.1	Transtypage [Notes en ligne]	114
9.9	Les variables [Notes en ligne]	115
9.9.1	Essentiel [Notes en ligne]	115
9.9.2	Variables prédéfinies [Notes en ligne]	116
9.9.2.1	Variables Apache [Notes en ligne]	116
9.9.2.2	Variables d'environnement [Notes en ligne]	118
9.9.2.3	Variables PHP [Notes en ligne]	118
9.9.3	Portée des variables [Notes en ligne]	119
9.9.4	Les variables dynamiques [Notes en ligne]	121
9.9.5	Variables externes à PHP [Notes en ligne]	121
9.9.5.1	Formulaires HTML (GET et POST) [Notes en ligne]	121
9.9.5.2	HTTP Cookies [Notes en ligne]	122
9.9.5.3	Variables d'environnement [Notes en ligne]	123
9.9.5.4	Cas des points dans les noms de variables [Notes en ligne]	123
9.9.5.5	Détermination du type des variables [Notes en ligne]	124
9.10	Les références [Notes en ligne]	124
9.10.1	Qu'est ce qu'une référence? [Notes en ligne]	124
9.10.2	Que font les références [Notes en ligne]	124
9.10.3	Ce que les références ne sont pas [Notes en ligne]	125
9.10.4	Passage par référence [Notes en ligne]	125
9.10.5	Retourner des références [Notes en ligne]	126
9.10.6	Détruire une références [Notes en ligne]	126
9.10.7	Repérer une référence [Notes en ligne]	127
9.10.7.1	Références global [Notes en ligne]	127
9.10.7.2	\$this [Notes en ligne]	127
10	Fonctions [Notes en ligne] [Exemples]	128
10.1	Apache [Notes en ligne]	128
10.1.1	apache_lookup_uri [Notes en ligne] [Exemples]	128
10.1.2	apache_note [Notes en ligne] [Exemples]	128
10.1.3	getallheaders [Notes en ligne] [Exemples]	129
10.1.4	virtual [Notes en ligne] [Exemples]	129
10.1.5	ascii2ebcdic [Notes en ligne] [Exemples]	129
10.1.6	ebcdic2ascii [Notes en ligne] [Exemples]	130
10.2	Tableaux [Notes en ligne]	130
10.2.1	array [Notes en ligne] [Exemples]	130
10.2.2	array_count_values [Notes en ligne] [Exemples]	131
10.2.3	array_diff [Notes en ligne] [Exemples]	132
10.2.4	array_flip [Notes en ligne] [Exemples]	132
10.2.5	array_intersect [Notes en ligne] [Exemples]	133

Table of Contents

10.2.6 array_keys [Notes en ligne] [Exemples]	133
10.2.7 array_merge [Notes en ligne] [Exemples]	134
10.2.8 array_merge_recursive [Notes en ligne] [Exemples]	134
10.2.9 array_multisort [Notes en ligne] [Exemples]	135
10.2.10 array_pad [Notes en ligne] [Exemples]	136
10.2.11 array_pop [Notes en ligne] [Exemples]	136
10.2.12 array_push [Notes en ligne] [Exemples]	137
10.2.13 array_reverse [Notes en ligne] [Exemples]	137
10.2.14 array_rand [Notes en ligne] [Exemples]	138
10.2.15 array_shift [Notes en ligne] [Exemples]	138
10.2.16 array_slice [Notes en ligne] [Exemples]	138
10.2.17 array_splice [Notes en ligne] [Exemples]	139
10.2.18 array_unique [Notes en ligne] [Exemples]	140
10.2.19 array_unshift [Notes en ligne] [Exemples]	140
10.2.20 array_values [Notes en ligne] [Exemples]	141
10.2.21 array_walk [Notes en ligne] [Exemples]	141
10.2.22 arsort [Notes en ligne] [Exemples]	142
10.2.23 asort [Notes en ligne] [Exemples]	142
10.2.24 compact [Notes en ligne] [Exemples]	143
10.2.25 count [Notes en ligne] [Exemples]	143
10.2.26 current [Notes en ligne] [Exemples]	144
10.2.27 each [Notes en ligne] [Exemples]	144
10.2.28 end [Notes en ligne] [Exemples]	145
10.2.29 extract [Notes en ligne] [Exemples]	146
10.2.30 in_array [Notes en ligne] [Exemples]	147
10.2.31 key [Notes en ligne] [Exemples]	147
10.2.32 krsort [Notes en ligne] [Exemples]	148
10.2.33 ksort [Notes en ligne] [Exemples]	148
10.2.34 list [Notes en ligne] [Exemples]	148
10.2.35 natsort [Notes en ligne] [Exemples]	149
10.2.36 natcasesort [Notes en ligne] [Exemples]	150
10.2.37 next [Notes en ligne] [Exemples]	150
10.2.38 pos [Notes en ligne] [Exemples]	150
10.2.39 prev [Notes en ligne] [Exemples]	151
10.2.40 range [Notes en ligne] [Exemples]	151
10.2.41 reset [Notes en ligne] [Exemples]	151
10.2.42 rsort [Notes en ligne] [Exemples]	151
10.2.43 shuffle [Notes en ligne] [Exemples]	152
10.2.44 sizeof [Notes en ligne] [Exemples]	152
10.2.45 sort [Notes en ligne] [Exemples]	152
10.2.46 uasort [Notes en ligne] [Exemples]	153
10.2.47 uksort [Notes en ligne] [Exemples]	153
10.2.48 usort [Notes en ligne] [Exemples]	153
10.3 Aspell [Notes en ligne]	154
10.3.1 aspell_new [Notes en ligne] [Exemples]	154
10.3.2 aspell_check [Notes en ligne] [Exemples]	155
10.3.3 aspell_check_raw [Notes en ligne] [Exemples]	155

Table of Contents

10.3.4	aspell_suggest [Notes en ligne] [Exemples].....	155
10.4	Nombres de grande taille [Notes en ligne]	156
10.4.1	bcadd [Notes en ligne] [Exemples].....	156
10.4.2	bccomp [Notes en ligne] [Exemples].....	156
10.4.3	bctdiv [Notes en ligne] [Exemples].....	156
10.4.4	bcmmod [Notes en ligne] [Exemples].....	157
10.4.5	bcmul [Notes en ligne] [Exemples].....	157
10.4.6	bcpow [Notes en ligne] [Exemples].....	157
10.4.7	bcscale [Notes en ligne] [Exemples].....	157
10.4.8	bcsqrt [Notes en ligne] [Exemples].....	158
10.4.9	bcsub [Notes en ligne] [Exemples].....	158
10.5	Calendrier [Notes en ligne]	158
10.5.1	jdtogregorian [Notes en ligne] [Exemples].....	158
10.5.2	gregoriantojd [Notes en ligne] [Exemples].....	159
10.5.3	jdtojulian [Notes en ligne] [Exemples].....	159
10.5.4	juliantojd [Notes en ligne] [Exemples].....	159
10.5.5	jdtojewish [Notes en ligne] [Exemples].....	159
10.5.6	jewishtojd [Notes en ligne] [Exemples].....	160
10.5.7	jdtofrrench [Notes en ligne] [Exemples].....	160
10.5.8	frenchtojd [Notes en ligne] [Exemples].....	160
10.5.9	jdmonthname [Notes en ligne] [Exemples].....	160
10.5.10	jddayofweek [Notes en ligne] [Exemples].....	161
10.5.11	easter_date [Notes en ligne] [Exemples].....	161
10.5.12	easter_days [Notes en ligne] [Exemples].....	162
10.5.13	unixtojd [Notes en ligne] [Exemples].....	162
10.5.14	jdtounix [Notes en ligne] [Exemples].....	162
10.6	Paiement CCVS [Notes en ligne]	163
10.7	Objets [Notes en ligne]	163
10.7.1	Introduction [Notes en ligne]	163
10.7.1.1	About [Notes en ligne]	163
10.7.1.2	Exemple d'utilisation [Notes en ligne]	163
10.7.2	get_declared_classes [Notes en ligne] [Exemples].....	165
10.7.3	call_user_method [Notes en ligne] [Exemples].....	165
10.7.4	class_exists [Notes en ligne] [Exemples].....	166
10.7.5	get_class [Notes en ligne] [Exemples].....	166
10.7.6	get_class_methods [Notes en ligne] [Exemples].....	166
10.7.7	get_class_vars [Notes en ligne] [Exemples].....	166
10.7.8	get_object_vars [Notes en ligne] [Exemples].....	167
10.7.9	get_parent_class [Notes en ligne] [Exemples].....	168
10.7.10	is_subclass_of [Notes en ligne] [Exemples].....	168
10.7.11	method_exists [Notes en ligne] [Exemples].....	168
10.8	Support COM pour Windows [Notes en ligne]	168
10.8.1	com_load [Notes en ligne] [Exemples].....	168
10.8.2	com_invoke [Notes en ligne] [Exemples].....	168
10.8.3	com_propget [Notes en ligne] [Exemples].....	169
10.8.4	com_get [Notes en ligne] [Exemples].....	169
10.8.5	com_propput [Notes en ligne] [Exemples].....	169

Table of Contents

10.8.6 com_propset [Notes en ligne] [Exemples]	169
10.8.7 com_set [Notes en ligne] [Exemples]	169
10.9 ClibPDF [Notes en ligne]	170
10.9.1 cpdf_global_set_document_limits [Notes en ligne] [Exemples]	172
10.9.2 cpdf_set_creator [Notes en ligne] [Exemples]	172
10.9.3 cpdf_set_title [Notes en ligne] [Exemples]	172
10.9.4 cpdf_set_subject [Notes en ligne] [Exemples]	172
10.9.5 cpdf_set_keywords [Notes en ligne] [Exemples]	173
10.9.6 cpdf_open [Notes en ligne] [Exemples]	173
10.9.7 cpdf_close [Notes en ligne] [Exemples]	173
10.9.8 cpdf_page_init [Notes en ligne] [Exemples]	173
10.9.9 cpdf_finalize_page [Notes en ligne] [Exemples]	174
10.9.10 cpdf_finalize [Notes en ligne] [Exemples]	174
10.9.11 cpdf_output_buffer [Notes en ligne] [Exemples]	174
10.9.12 cpdf_save_to_file [Notes en ligne] [Exemples]	174
10.9.13 cpdf_set_current_page [Notes en ligne] [Exemples]	174
10.9.14 cpdf_begin_text [Notes en ligne] [Exemples]	175
10.9.15 cpdf_end_text [Notes en ligne] [Exemples]	175
10.9.16 cpdf_show [Notes en ligne] [Exemples]	175
10.9.17 cpdf_show_xy [Notes en ligne] [Exemples]	176
10.9.18 cpdf_text [Notes en ligne] [Exemples]	176
10.9.19 cpdf_set_font [Notes en ligne] [Exemples]	176
10.9.20 cpdf_set_leading [Notes en ligne] [Exemples]	176
10.9.21 cpdf_set_text_rendering [Notes en ligne] [Exemples]	177
10.9.22 cpdf_set_horiz_scaling [Notes en ligne] [Exemples]	177
10.9.23 cpdf_set_text_rise [Notes en ligne] [Exemples]	177
10.9.24 cpdf_set_text_matrix [Notes en ligne] [Exemples]	177
10.9.25 cpdf_set_text_pos [Notes en ligne] [Exemples]	177
10.9.26 cpdf_set_char_spacing [Notes en ligne] [Exemples]	178
10.9.27 cpdf_set_word_spacing [Notes en ligne] [Exemples]	178
10.9.28 cpdf_continue_text [Notes en ligne] [Exemples]	178
10.9.29 cpdf_stringwidth [Notes en ligne] [Exemples]	178
10.9.30 cpdf_save [Notes en ligne] [Exemples]	178
10.9.31 cpdf_restore [Notes en ligne] [Exemples]	179
10.9.32 cpdf_translate [Notes en ligne] [Exemples]	179
10.9.33 cpdf_scale [Notes en ligne] [Exemples]	179
10.9.34 cpdf_rotate [Notes en ligne] [Exemples]	179
10.9.35 cpdf_setflat [Notes en ligne] [Exemples]	180
10.9.36 cpdf_setlinejoin [Notes en ligne] [Exemples]	180
10.9.37 cpdf_setlinecap [Notes en ligne] [Exemples]	180
10.9.38 cpdf_setmiterlimit [Notes en ligne] [Exemples]	180
10.9.39 cpdf_setlinewidth [Notes en ligne] [Exemples]	180
10.9.40 cpdf_setdash [Notes en ligne] [Exemples]	180
10.9.41 cpdf_newpath [Notes en ligne] [Exemples]	181
10.9.42 cpdf_moveto [Notes en ligne] [Exemples]	181
10.9.43 cpdf_rmoveto [Notes en ligne] [Exemples]	181
10.9.44 cpdf_curveto [Notes en ligne] [Exemples]	181

Table of Contents

10.9.45	cpdf_lineto [Notes en ligne] [Exemples]	182
10.9.46	cpdf_rlineto [Notes en ligne] [Exemples]	182
10.9.47	cpdf_circle [Notes en ligne] [Exemples]	182
10.9.48	cpdf_arc [Notes en ligne] [Exemples]	182
10.9.49	cpdf_rect [Notes en ligne] [Exemples]	183
10.9.50	cpdf_closepath [Notes en ligne] [Exemples]	183
10.9.51	cpdf_stroke [Notes en ligne] [Exemples]	183
10.9.52	cpdf_closepath_stroke [Notes en ligne] [Exemples]	183
10.9.53	cpdf_fill [Notes en ligne] [Exemples]	183
10.9.54	cpdf_fill_stroke [Notes en ligne] [Exemples]	184
10.9.55	cpdf_closepath_fill_stroke [Notes en ligne] [Exemples]	184
10.9.56	cpdf_clip [Notes en ligne] [Exemples]	184
10.9.57	cpdf_setgray_fill [Notes en ligne] [Exemples]	184
10.9.58	cpdf_setgray_stroke [Notes en ligne] [Exemples]	184
10.9.59	cpdf_setgray [Notes en ligne] [Exemples]	185
10.9.60	cpdf_setrgbcolor_fill [Notes en ligne] [Exemples]	185
10.9.61	cpdf_setrgbcolor_stroke [Notes en ligne] [Exemples]	185
10.9.62	cpdf_setrgbcolor [Notes en ligne] [Exemples]	185
10.9.63	cpdf_add_outline [Notes en ligne] [Exemples]	185
10.9.64	cpdf_set_page_animation [Notes en ligne] [Exemples]	186
10.9.65	cpdf_import_jpeg [Notes en ligne] [Exemples]	186
10.9.66	cpdf_place_inline_image [Notes en ligne] [Exemples]	187
10.9.67	cpdf_add_annotation [Notes en ligne] [Exemples]	187
10.10	CURL [Notes en ligne]	187
10.10.1	curl_init [Notes en ligne] [Exemples]	188
10.10.2	curl_setopt [Notes en ligne] [Exemples]	188
10.10.3	curl_exec [Notes en ligne] [Exemples]	190
10.10.4	curl_close [Notes en ligne] [Exemples]	191
10.10.5	curl_version [Notes en ligne] [Exemples]	191
10.11	Paiement Cybercash [Notes en ligne]	191
10.11.1	cybercash_encr [Notes en ligne] [Exemples]	191
10.11.2	cybercash_decr [Notes en ligne] [Exemples]	191
10.11.3	cybercash_base64_encode [Notes en ligne] [Exemples]	192
10.11.4	cybercash_base64_decode [Notes en ligne] [Exemples]	192
10.12	Dates et heures [Notes en ligne]	192
10.12.1	checkdate [Notes en ligne] [Exemples]	192
10.12.2	date [Notes en ligne] [Exemples]	192
10.12.3	getdate [Notes en ligne] [Exemples]	194
10.12.4	gettimeofday [Notes en ligne] [Exemples]	194
10.12.5	gmdate [Notes en ligne] [Exemples]	194
10.12.6	gmmktime [Notes en ligne] [Exemples]	195
10.12.7	gmstrftime [Notes en ligne] [Exemples]	195
10.12.8	localtime [Notes en ligne] [Exemples]	195
10.12.9	microtime [Notes en ligne] [Exemples]	196
10.12.10	mktime [Notes en ligne] [Exemples]	196
10.12.11	strftime [Notes en ligne] [Exemples]	197
10.12.12	time [Notes en ligne] [Exemples]	198

Table of Contents

10.12.13	strtotime [Notes en ligne] [Exemples]	198
10.13	DBA [Notes en ligne]	199
10.13.1	dba_close [Notes en ligne] [Exemples]	200
10.13.2	dba_delete [Notes en ligne] [Exemples]	200
10.13.3	dba_exists [Notes en ligne] [Exemples]	201
10.13.4	dba_fetch [Notes en ligne] [Exemples]	201
10.13.5	dba_firstkey [Notes en ligne] [Exemples]	201
10.13.6	dba_insert [Notes en ligne] [Exemples]	201
10.13.7	dba_nextkey [Notes en ligne] [Exemples]	202
10.13.8	dba_popen [Notes en ligne] [Exemples]	202
10.13.9	dba_open [Notes en ligne] [Exemples]	202
10.13.10	dba_optimize [Notes en ligne] [Exemples]	202
10.13.11	dba_replace [Notes en ligne] [Exemples]	203
10.13.12	dba_sync [Notes en ligne] [Exemples]	203
10.14	dBase [Notes en ligne]	203
10.14.1	dbase_create [Notes en ligne] [Exemples]	204
10.14.2	dbase_open [Notes en ligne] [Exemples]	205
10.14.3	dbase_close [Notes en ligne] [Exemples]	205
10.14.4	dbase_pack [Notes en ligne] [Exemples]	205
10.14.5	dbase_add_record [Notes en ligne] [Exemples]	205
10.14.6	dbase_replace_record [Notes en ligne] [Exemples]	205
10.14.7	dbase_delete_record [Notes en ligne] [Exemples]	206
10.14.8	dbase_get_record [Notes en ligne] [Exemples]	206
10.14.9	dbase_get_record_with_names [Notes en ligne] [Exemples]	206
10.14.10	dbase_numfields [Notes en ligne] [Exemples]	206
10.14.11	dbase_numrecords [Notes en ligne] [Exemples]	207
10.15	DBM [Notes en ligne]	207
10.15.1	dbmopen [Notes en ligne] [Exemples]	207
10.15.2	dbmclose [Notes en ligne] [Exemples]	208
10.15.3	dbmexists [Notes en ligne] [Exemples]	208
10.15.4	dbmfetch [Notes en ligne] [Exemples]	208
10.15.5	dbminsert [Notes en ligne] [Exemples]	208
10.15.6	dbmreplace [Notes en ligne] [Exemples]	208
10.15.7	dbmdelete [Notes en ligne] [Exemples]	209
10.15.8	dbmfirstkey [Notes en ligne] [Exemples]	209
10.15.9	dbmnextkey [Notes en ligne] [Exemples]	209
10.15.10	dblist [Notes en ligne] [Exemples]	209
10.16	Accès aux dossiers [Notes en ligne]	210
10.16.1	chdir [Notes en ligne] [Exemples]	210
10.16.2	dir [Notes en ligne] [Exemples]	210
10.16.3	closedir [Notes en ligne] [Exemples]	210
10.16.4	getcwd [Notes en ligne] [Exemples]	210
10.16.5	opendir [Notes en ligne] [Exemples]	211
10.16.6	readdir [Notes en ligne] [Exemples]	211
10.16.7	rewinddir [Notes en ligne] [Exemples]	212
10.17	DOM XML [Notes en ligne]	212
10.17.1	xmldoc [Notes en ligne] [Exemples]	212

Table of Contents

10.17.2	xmldocfile [Notes en ligne] [Exemples]	213
10.17.3	xmltree [Notes en ligne] [Exemples]	213
10.18	Gestion des erreurs [Notes en ligne]	213
10.18.1	error_log [Notes en ligne] [Exemples]	213
10.18.2	error_reporting [Notes en ligne] [Exemples]	214
10.18.3	restore_error_handler [Notes en ligne] [Exemples]	215
10.18.4	set_error_handler [Notes en ligne] [Exemples]	215
10.18.5	trigger_error [Notes en ligne] [Exemples]	218
10.18.6	user_error [Notes en ligne] [Exemples]	218
10.19	Exécution de programmes externes [Notes en ligne]	218
10.19.1	escapeshellarg [Notes en ligne] [Exemples]	218
10.19.2	escapeshellcmd [Notes en ligne] [Exemples]	219
10.19.3	exec [Notes en ligne] [Exemples]	219
10.19.4	passthru [Notes en ligne] [Exemples]	219
10.19.5	system [Notes en ligne] [Exemples]	220
10.20	Forms Data Format [Notes en ligne]	220
10.20.1	fdf_open [Notes en ligne] [Exemples]	221
10.20.2	fdf_close [Notes en ligne] [Exemples]	222
10.20.3	fdf_create [Notes en ligne] [Exemples]	222
10.20.4	fdf_save [Notes en ligne] [Exemples]	222
10.20.5	fdf_get_value [Notes en ligne] [Exemples]	223
10.20.6	fdf_set_value [Notes en ligne] [Exemples]	223
10.20.7	fdf_next_field_name [Notes en ligne] [Exemples]	223
10.20.8	fdf_set_ap [Notes en ligne] [Exemples]	223
10.20.9	fdf_set_status [Notes en ligne] [Exemples]	223
10.20.10	fdf_get_status [Notes en ligne] [Exemples]	224
10.20.11	fdf_set_file [Notes en ligne] [Exemples]	224
10.20.12	fdf_get_file [Notes en ligne] [Exemples]	224
10.20.13	fdf_set_flags [Notes en ligne] [Exemples]	224
10.20.14	fdf_set_opt [Notes en ligne] [Exemples]	224
10.20.15	fdf_set_submit_form_action [Notes en ligne] [Exemples]	225
10.20.16	fdf_set_javascript_action [Notes en ligne] [Exemples]	225
10.21	FilePro [Notes en ligne]	225
10.21.1	filepro [Notes en ligne] [Exemples]	225
10.21.2	filepro_fieldname [Notes en ligne] [Exemples]	225
10.21.3	filepro_fieldtype [Notes en ligne] [Exemples]	226
10.21.4	filepro_fieldwidth [Notes en ligne] [Exemples]	226
10.21.5	filepro_retrieve [Notes en ligne] [Exemples]	226
10.21.6	filepro_fieldcount [Notes en ligne] [Exemples]	226
10.21.7	filepro_rowcount [Notes en ligne] [Exemples]	226
10.22	Système de fichiers [Notes en ligne]	227
10.22.1	basename [Notes en ligne] [Exemples]	227
10.22.2	chgrp [Notes en ligne] [Exemples]	227
10.22.3	chmod [Notes en ligne] [Exemples]	227
10.22.4	chown [Notes en ligne] [Exemples]	228
10.22.5	clearstatcache [Notes en ligne] [Exemples]	228
10.22.6	copy [Notes en ligne] [Exemples]	228

Table of Contents

10.22.7 delete [Notes en ligne] [Exemples]	229
10.22.8 dirname [Notes en ligne] [Exemples]	229
10.22.9 diskfree [Notes en ligne] [Exemples]	229
10.22.10 fclose [Notes en ligne] [Exemples]	229
10.22.11 feof [Notes en ligne] [Exemples]	230
10.22.12 fflush [Notes en ligne] [Exemples]	230
10.22.13 fgetc [Notes en ligne] [Exemples]	230
10.22.14 fgetcsv [Notes en ligne] [Exemples]	230
10.22.15 fgets [Notes en ligne] [Exemples]	231
10.22.16 fgetss [Notes en ligne] [Exemples]	231
10.22.17 file [Notes en ligne] [Exemples]	232
10.22.18 file_exists [Notes en ligne] [Exemples]	232
10.22.19 fileatime [Notes en ligne] [Exemples]	232
10.22.20 filectime [Notes en ligne] [Exemples]	233
10.22.21 filegroup [Notes en ligne] [Exemples]	233
10.22.22 fileinode [Notes en ligne] [Exemples]	233
10.22.23 filemtime [Notes en ligne] [Exemples]	233
10.22.24 fileowner [Notes en ligne] [Exemples]	234
10.22.25 fileperms [Notes en ligne] [Exemples]	234
10.22.26 filesize [Notes en ligne] [Exemples]	234
10.22.27 filetype [Notes en ligne] [Exemples]	234
10.22.28 flock [Notes en ligne] [Exemples]	234
10.22.29 fopen [Notes en ligne] [Exemples]	235
10.22.30 fpassthru [Notes en ligne] [Exemples]	236
10.22.31 fputs [Notes en ligne] [Exemples]	236
10.22.32 fread [Notes en ligne] [Exemples]	237
10.22.33 fscanf [Notes en ligne] [Exemples]	237
10.22.34 fseek [Notes en ligne] [Exemples]	238
10.22.35 fstat [Notes en ligne] [Exemples]	238
10.22.36 ftell [Notes en ligne] [Exemples]	239
10.22.37 ftruncate [Notes en ligne] [Exemples]	239
10.22.38 fwrite [Notes en ligne] [Exemples]	239
10.22.39 set_file_buffer [Notes en ligne] [Exemples]	239
10.22.40 is_dir [Notes en ligne] [Exemples]	240
10.22.41 is_executable [Notes en ligne] [Exemples]	240
10.22.42 is_file [Notes en ligne] [Exemples]	240
10.22.43 is_link [Notes en ligne] [Exemples]	240
10.22.44 is_readable [Notes en ligne] [Exemples]	241
10.22.45 is_writable [Notes en ligne] [Exemples]	241
10.22.46 is_uploaded_file [Notes en ligne] [Exemples]	241
10.22.47 link [Notes en ligne] [Exemples]	241
10.22.48 linkinfo [Notes en ligne] [Exemples]	242
10.22.49 mkdir [Notes en ligne] [Exemples]	242
10.22.50 move_uploaded_file [Notes en ligne] [Exemples]	242
10.22.51 pclose [Notes en ligne] [Exemples]	243
10.22.52 popen [Notes en ligne] [Exemples]	243
10.22.53 readfile [Notes en ligne] [Exemples]	243

Table of Contents

10.22.54	readlink [Notes en ligne] [Exemples]	244
10.22.55	rename [Notes en ligne] [Exemples]	244
10.22.56	rewind [Notes en ligne] [Exemples]	244
10.22.57	rmdir [Notes en ligne] [Exemples]	244
10.22.58	stat [Notes en ligne] [Exemples]	245
10.22.59	lstat [Notes en ligne] [Exemples]	245
10.22.60	realpath [Notes en ligne] [Exemples]	246
10.22.61	symlink [Notes en ligne] [Exemples]	246
10.22.62	tempnam [Notes en ligne] [Exemples]	246
10.22.63	tmpfile [Notes en ligne] [Exemples]	247
10.22.64	touch [Notes en ligne] [Exemples]	247
10.22.65	umask [Notes en ligne] [Exemples]	247
10.22.66	unlink [Notes en ligne] [Exemples]	248
10.23	FTP [Notes en ligne]	248
10.23.1	ftp_connect [Notes en ligne] [Exemples]	248
10.23.2	ftp_login [Notes en ligne] [Exemples]	249
10.23.3	ftp_pwd [Notes en ligne] [Exemples]	249
10.23.4	ftp_cdup [Notes en ligne] [Exemples]	249
10.23.5	ftp_chdir [Notes en ligne] [Exemples]	249
10.23.6	ftp_mkdir [Notes en ligne] [Exemples]	249
10.23.7	ftp_rmdir [Notes en ligne] [Exemples]	250
10.23.8	ftp_nlist [Notes en ligne] [Exemples]	250
10.23.9	ftp_rawlist [Notes en ligne] [Exemples]	250
10.23.10	ftp_systype [Notes en ligne] [Exemples]	250
10.23.11	ftp_pasv [Notes en ligne] [Exemples]	250
10.23.12	ftp_get [Notes en ligne] [Exemples]	251
10.23.13	ftp_fget [Notes en ligne] [Exemples]	251
10.23.14	ftp_put [Notes en ligne] [Exemples]	251
10.23.15	ftp_fput [Notes en ligne] [Exemples]	251
10.23.16	ftp_size [Notes en ligne] [Exemples]	251
10.23.17	ftp_mdtm [Notes en ligne] [Exemples]	252
10.23.18	ftp_rename [Notes en ligne] [Exemples]	252
10.23.19	ftp_delete [Notes en ligne] [Exemples]	252
10.23.20	ftp_site [Notes en ligne] [Exemples]	252
10.23.21	ftp_quit [Notes en ligne] [Exemples]	253
10.24	Fonctions [Notes en ligne]	253
10.24.1	call_user_func [Notes en ligne] [Exemples]	253
10.24.2	create_function [Notes en ligne] [Exemples]	253
10.24.3	func_get_arg [Notes en ligne] [Exemples]	255
10.24.4	func_get_args [Notes en ligne] [Exemples]	256
10.24.5	func_num_args [Notes en ligne] [Exemples]	256
10.24.6	function_exists [Notes en ligne] [Exemples]	257
10.24.7	register_shutdown_function [Notes en ligne] [Exemples]	257
10.25	GNU Gettext [Notes en ligne]	257
10.25.1	bindtextdomain [Notes en ligne] [Exemples]	257
10.25.2	dcgettext [Notes en ligne] [Exemples]	258
10.25.3	dgettext [Notes en ligne] [Exemples]	258

Table of Contents

10.25.4	gettext [Notes en ligne] [Exemples].....	258
10.25.5	textdomain [Notes en ligne] [Exemples].....	258
10.26	GMP [Notes en ligne]	259
10.26.1	gmp_init [Notes en ligne] [Exemples].....	259
10.26.2	gmp_intval [Notes en ligne] [Exemples].....	260
10.26.3	gmp_strval [Notes en ligne] [Exemples].....	260
10.26.4	gmp_add [Notes en ligne] [Exemples].....	260
10.26.5	gmp_sub [Notes en ligne] [Exemples].....	260
10.26.6	gmp_mul [Notes en ligne] [Exemples].....	260
10.26.7	gmp_div_q [Notes en ligne] [Exemples].....	261
10.26.8	gmp_div_r [Notes en ligne] [Exemples].....	261
10.26.9	gmp_div_qr [Notes en ligne] [Exemples].....	261
10.26.10	gmp_div [Notes en ligne] [Exemples].....	261
10.26.11	gmp_mod [Notes en ligne] [Exemples].....	262
10.26.12	gmp_divexact [Notes en ligne] [Exemples].....	262
10.26.13	gmp_cmp [Notes en ligne] [Exemples].....	262
10.26.14	gmp_neg [Notes en ligne] [Exemples].....	262
10.26.15	gmp_abs [Notes en ligne] [Exemples].....	262
10.26.16	gmp_sign [Notes en ligne] [Exemples].....	262
10.26.17	gmp_fact [Notes en ligne] [Exemples].....	263
10.26.18	gmp_sqrt [Notes en ligne] [Exemples].....	263
10.26.19	gmp_sqrtrem [Notes en ligne] [Exemples].....	263
10.26.20	gmp_perfect_square [Notes en ligne] [Exemples].....	263
10.26.21	gmp_pow [Notes en ligne] [Exemples].....	263
10.26.22	gmp_powm [Notes en ligne] [Exemples].....	263
10.26.23	gmp_prob_prime [Notes en ligne] [Exemples].....	264
10.26.24	gmp_gcd [Notes en ligne] [Exemples].....	264
10.26.25	gmp_gcdext [Notes en ligne] [Exemples].....	264
10.26.26	gmp_invert [Notes en ligne] [Exemples].....	264
10.26.27	gmp_legendre [Notes en ligne] [Exemples].....	264
10.26.28	gmp_jacobi [Notes en ligne] [Exemples].....	264
10.26.29	gmp_random [Notes en ligne] [Exemples].....	265
10.26.30	gmp_and [Notes en ligne] [Exemples].....	265
10.26.31	gmp_or [Notes en ligne] [Exemples].....	265
10.26.32	gmp_xor [Notes en ligne] [Exemples].....	265
10.26.33	gmp_setbit [Notes en ligne] [Exemples].....	265
10.26.34	gmp_clrbit [Notes en ligne] [Exemples].....	265
10.26.35	gmp_scan0 [Notes en ligne] [Exemples].....	266
10.26.36	gmp_scan1 [Notes en ligne] [Exemples].....	266
10.26.37	gmp_popcount [Notes en ligne] [Exemples].....	266
10.26.38	gmp_hamdist [Notes en ligne] [Exemples].....	266
10.27	HTTP [Notes en ligne]	266
10.27.1	header [Notes en ligne] [Exemples].....	266
10.27.2	headers_sent [Notes en ligne] [Exemples].....	267
10.27.3	setcookie [Notes en ligne] [Exemples].....	267
10.28	Hyperwave [Notes en ligne]	268
10.28.1	Introduction [Notes en ligne]	269

Table of Contents

10.28.2 Intégration avec Apache [Notes en ligne]	271
10.28.3 A faire [Notes en ligne]	272
10.28.4 hw_array2objrec [Notes en ligne] [Exemples]	272
10.28.5 hw_children [Notes en ligne] [Exemples]	272
10.28.6 hw_childrenobj [Notes en ligne] [Exemples]	272
10.28.7 hw_close [Notes en ligne] [Exemples]	273
10.28.8 hw_connect [Notes en ligne] [Exemples]	273
10.28.9 hw_cp [Notes en ligne] [Exemples]	273
10.28.10 hw_deleteobject [Notes en ligne] [Exemples]	273
10.28.11 hw_dochyanchor [Notes en ligne] [Exemples]	274
10.28.12 hw_dochyanchorobj [Notes en ligne] [Exemples]	274
10.28.13 hw_documentattributes [Notes en ligne] [Exemples]	274
10.28.14 hw_documentbodytag [Notes en ligne] [Exemples]	274
10.28.15 hw_documentcontent [Notes en ligne] [Exemples]	274
10.28.16 hw_documentsetcontent [Notes en ligne] [Exemples]	275
10.28.17 hw_documentsize [Notes en ligne] [Exemples]	275
10.28.18 hw_errormsg [Notes en ligne] [Exemples]	275
10.28.19 hw_edittest [Notes en ligne] [Exemples]	275
10.28.20 hw_error [Notes en ligne] [Exemples]	275
10.28.21 hw_free_document [Notes en ligne] [Exemples]	276
10.28.22 hw_getparents [Notes en ligne] [Exemples]	276
10.28.23 hw_getparentsobj [Notes en ligne] [Exemples]	276
10.28.24 hw_getchildcoll [Notes en ligne] [Exemples]	276
10.28.25 hw_getchildcollobj [Notes en ligne] [Exemples]	276
10.28.26 hw_getremote [Notes en ligne] [Exemples]	277
10.28.27 hw_getremotechildren [Notes en ligne] [Exemples]	277
10.28.28 hw_getsrcbydestobj [Notes en ligne] [Exemples]	277
10.28.29 hw_getobject [Notes en ligne] [Exemples]	278
10.28.30 hw_getandlock [Notes en ligne] [Exemples]	278
10.28.31 hw_gettext [Notes en ligne] [Exemples]	278
10.28.32 hw_getobjectbyquery [Notes en ligne] [Exemples]	279
10.28.33 hw_getobjectbyqueryobj [Notes en ligne] [Exemples]	279
10.28.34 hw_getobjectbyquerycoll [Notes en ligne] [Exemples]	279
10.28.35 hw_getobjectbyquerycollobj [Notes en ligne] [Exemples]	280
10.28.36 hw_getchilddoccoll [Notes en ligne] [Exemples]	280
10.28.37 hw_getchilddoccollobj [Notes en ligne] [Exemples]	280
10.28.38 hw_getanchors [Notes en ligne] [Exemples]	280
10.28.39 hw_getanchorsobj [Notes en ligne] [Exemples]	280
10.28.40 hw_mv [Notes en ligne] [Exemples]	281
10.28.41 hw_identify [Notes en ligne] [Exemples]	281
10.28.42 hw_incollections [Notes en ligne] [Exemples]	281
10.28.43 hw_info [Notes en ligne] [Exemples]	281
10.28.44 hw_inscoll [Notes en ligne] [Exemples]	282
10.28.45 hw_insdock [Notes en ligne] [Exemples]	282
10.28.46 hw_insertdocument [Notes en ligne] [Exemples]	282
10.28.47 hw_insertobject [Notes en ligne] [Exemples]	282
10.28.48 hw_mapid [Notes en ligne] [Exemples]	283

Table of Contents

10.28.49	hw_modifyobject [Notes en ligne] [Exemples]	283
10.28.50	hw_new_document [Notes en ligne] [Exemples]	285
10.28.51	hw_objrec2array [Notes en ligne] [Exemples]	285
10.28.52	hw_outputdocument [Notes en ligne] [Exemples]	285
10.28.53	hw_pconnect [Notes en ligne] [Exemples]	285
10.28.54	hw_pipedocument [Notes en ligne] [Exemples]	286
10.28.55	hw_root [Notes en ligne] [Exemples]	286
10.28.56	hw_unlock [Notes en ligne] [Exemples]	286
10.28.57	hw_who [Notes en ligne] [Exemples]	286
10.28.58	hw_username [Notes en ligne] [Exemples]	286
10.29	Fonctions InterBase [Notes en ligne]	287
10.29.1	ibase_connect [Notes en ligne] [Exemples]	287
10.29.2	ibase_pconnect [Notes en ligne] [Exemples]	288
10.29.3	ibase_close [Notes en ligne] [Exemples]	288
10.29.4	ibase_query [Notes en ligne] [Exemples]	288
10.29.5	ibase_fetch_row [Notes en ligne] [Exemples]	289
10.29.6	ibase_fetch_object [Notes en ligne] [Exemples]	289
10.29.7	ibase_field_info [Notes en ligne] [Exemples]	289
10.29.8	ibase_free_result [Notes en ligne] [Exemples]	290
10.29.9	ibase_prepare [Notes en ligne] [Exemples]	290
10.29.10	ibase_execute [Notes en ligne] [Exemples]	290
10.29.11	ibase_trans [Notes en ligne] [Exemples]	290
10.29.12	ibase_commit [Notes en ligne] [Exemples]	291
10.29.13	ibase_rollback [Notes en ligne] [Exemples]	291
10.29.14	ibase_free_query [Notes en ligne] [Exemples]	291
10.29.15	ibase_timefmt [Notes en ligne] [Exemples]	291
10.29.16	ibase_num_fields [Notes en ligne] [Exemples]	292
10.29.17	ibase_errmsg [Notes en ligne] [Exemples]	292
10.30	ICAP [Notes en ligne]	292
10.30.1	icap_open [Notes en ligne] [Exemples]	292
10.30.2	icap_close [Notes en ligne] [Exemples]	293
10.30.3	icap_fetch_event [Notes en ligne] [Exemples]	293
10.30.4	icap_list_events [Notes en ligne] [Exemples]	293
10.30.5	icap_store_event [Notes en ligne] [Exemples]	294
10.30.6	icap_delete_event [Notes en ligne] [Exemples]	294
10.30.7	icap_snooze [Notes en ligne] [Exemples]	295
10.30.8	icap_list_alarms [Notes en ligne] [Exemples]	295
10.31	Informix [Notes en ligne]	295
10.31.1	ifx_connect [Notes en ligne] [Exemples]	297
10.31.2	ifx_pconnect [Notes en ligne] [Exemples]	297
10.31.3	ifx_close [Notes en ligne] [Exemples]	297
10.31.4	ifx_query [Notes en ligne] [Exemples]	298
10.31.5	ifx_prepare [Notes en ligne] [Exemples]	299
10.31.6	ifx_do [Notes en ligne] [Exemples]	300
10.31.7	ifx_error [Notes en ligne] [Exemples]	300
10.31.8	ifx_errormsg [Notes en ligne] [Exemples]	300
10.31.9	ifx_affected_rows [Notes en ligne] [Exemples]	301

Table of Contents

10.31.10 ifx_getsqlca [Notes en ligne] [Exemples]	301
10.31.11 ifx_fetch_row [Notes en ligne] [Exemples]	302
10.31.12 ifx_htmltbl_result [Notes en ligne] [Exemples]	303
10.31.13 ifx_fieldtypes [Notes en ligne] [Exemples]	303
10.31.14 ifx_fieldproperties [Notes en ligne] [Exemples]	304
10.31.15 ifx_num_fields [Notes en ligne] [Exemples]	304
10.31.16 ifx_num_rows [Notes en ligne] [Exemples]	304
10.31.17 ifx_free_result [Notes en ligne] [Exemples]	304
10.31.18 ifx_create_char [Notes en ligne] [Exemples]	305
10.31.19 ifx_free_char [Notes en ligne] [Exemples]	305
10.31.20 ifx_update_char [Notes en ligne] [Exemples]	305
10.31.21 ifx_get_char [Notes en ligne] [Exemples]	305
10.31.22 ifx_create_blob [Notes en ligne] [Exemples]	305
10.31.23 ifx_copy_blob [Notes en ligne] [Exemples]	306
10.31.24 ifx_free_blob [Notes en ligne] [Exemples]	306
10.31.25 ifx_get_blob [Notes en ligne] [Exemples]	306
10.31.26 ifx_update_blob [Notes en ligne] [Exemples]	306
10.31.27 ifx_blobinfile_mode [Notes en ligne] [Exemples]	306
10.31.28 ifx_textasvarchar [Notes en ligne] [Exemples]	306
10.31.29 ifx_byteasvarchar [Notes en ligne] [Exemples]	307
10.31.30 ifx_nullformat [Notes en ligne] [Exemples]	307
10.31.31 ifxus_create_slob [Notes en ligne] [Exemples]	307
10.31.32 ifx_free_slob [Notes en ligne] [Exemples]	307
10.31.33 ifxus_close_slob [Notes en ligne] [Exemples]	307
10.31.34 ifxus_open_slob [Notes en ligne] [Exemples]	308
10.31.35 ifxus_tell_slob [Notes en ligne] [Exemples]	308
10.31.36 ifxus_seek_slob [Notes en ligne] [Exemples]	308
10.31.37 ifxus_read_slob [Notes en ligne] [Exemples]	308
10.31.38 ifxus_write_slob [Notes en ligne] [Exemples]	309
10.32 Images [Notes en ligne]	309
10.32.1 getimagesize [Notes en ligne] [Exemples]	309
10.32.2 imagearc [Notes en ligne] [Exemples]	310
10.32.3 imagechar [Notes en ligne] [Exemples]	310
10.32.4 imagecharup [Notes en ligne] [Exemples]	310
10.32.5 imagecolorallocate [Notes en ligne] [Exemples]	311
10.32.6 imagecolordeallocate [Notes en ligne] [Exemples]	311
10.32.7 imagecolorat [Notes en ligne] [Exemples]	311
10.32.8 imagecolorclosest [Notes en ligne] [Exemples]	311
10.32.9 imagecolorexact [Notes en ligne] [Exemples]	312
10.32.10 imagecolorresolve [Notes en ligne] [Exemples]	312
10.32.11 imagegammacorrect [Notes en ligne] [Exemples]	312
10.32.12 imagecolorset [Notes en ligne] [Exemples]	312
10.32.13 imagecolorsforindex [Notes en ligne] [Exemples]	313
10.32.14 imagecolorstotal [Notes en ligne] [Exemples]	313
10.32.15 imagecolortransparent [Notes en ligne] [Exemples]	313
10.32.16 imagecopy [Notes en ligne] [Exemples]	313
10.32.17 imagecopyresized [Notes en ligne] [Exemples]	313

Table of Contents

10.32.18	imagecreate [Notes en ligne] [Exemples].....	314
10.32.19	imagecreatefromgif [Notes en ligne] [Exemples].....	314
10.32.20	imagecreatefromjpeg [Notes en ligne] [Exemples].....	315
10.32.21	imagecreatefrompng [Notes en ligne] [Exemples].....	315
10.32.22	imagecreatefromwbmp [Notes en ligne] [Exemples].....	316
10.32.23	imagecreatefromstring [Notes en ligne] [Exemples].....	316
10.32.24	imagedashedline [Notes en ligne] [Exemples].....	316
10.32.25	imagedestroy [Notes en ligne] [Exemples].....	317
10.32.26	imagefill [Notes en ligne] [Exemples].....	317
10.32.27	imagefilledpolygon [Notes en ligne] [Exemples].....	317
10.32.28	imagefilledrectangle [Notes en ligne] [Exemples].....	317
10.32.29	imagefilltoborder [Notes en ligne] [Exemples].....	317
10.32.30	imagefontheight [Notes en ligne] [Exemples].....	318
10.32.31	imagefontwidth [Notes en ligne] [Exemples].....	318
10.32.32	imagegif [Notes en ligne] [Exemples].....	318
10.32.33	imagepng [Notes en ligne] [Exemples].....	319
10.32.34	imagejpeg [Notes en ligne] [Exemples].....	320
10.32.35	imagewbmp [Notes en ligne] [Exemples].....	320
10.32.36	imageinterlace [Notes en ligne] [Exemples].....	320
10.32.37	imageline [Notes en ligne] [Exemples].....	320
10.32.38	imageloadfont [Notes en ligne] [Exemples].....	321
10.32.39	imagepolygon [Notes en ligne] [Exemples].....	321
10.32.40	imagepsbbox [Notes en ligne] [Exemples].....	321
10.32.41	imagepsencodefont [Notes en ligne] [Exemples].....	322
10.32.42	imagepsfreefont [Notes en ligne] [Exemples].....	322
10.32.43	imagepsloadfont [Notes en ligne] [Exemples].....	322
10.32.44	imagepsextendfont [Notes en ligne] [Exemples].....	323
10.32.45	imagepslantfont [Notes en ligne] [Exemples].....	323
10.32.46	imagepstext [Notes en ligne] [Exemples].....	323
10.32.47	imagerectangle [Notes en ligne] [Exemples].....	324
10.32.48	imagesetpixel [Notes en ligne] [Exemples].....	324
10.32.49	imagestring [Notes en ligne] [Exemples].....	324
10.32.50	imagestringup [Notes en ligne] [Exemples].....	324
10.32.51	imagesx [Notes en ligne] [Exemples].....	325
10.32.52	imagesy [Notes en ligne] [Exemples].....	325
10.32.53	imageftbbox [Notes en ligne] [Exemples].....	325
10.32.54	imagefttext [Notes en ligne] [Exemples].....	326
10.32.55	imagetypes [Notes en ligne] [Exemples].....	327
10.32.56	read_exif_data [Notes en ligne] [Exemples].....	327
10.33	IMAP [Notes en ligne].....	328
10.33.1	imap_append [Notes en ligne] [Exemples].....	328
10.33.2	imap_base64 [Notes en ligne] [Exemples].....	329
10.33.3	imap_body [Notes en ligne] [Exemples].....	329
10.33.4	imap_check [Notes en ligne] [Exemples].....	329
10.33.5	imap_close [Notes en ligne] [Exemples].....	330
10.33.6	imap_createmailbox [Notes en ligne] [Exemples].....	330
10.33.7	imap_delete [Notes en ligne] [Exemples].....	331

Table of Contents

10.33.8 imap_deletemailbox [Notes en ligne] [Exemples]	331
10.33.9 imap_expunge [Notes en ligne] [Exemples]	332
10.33.10 imap_fetchbody [Notes en ligne] [Exemples]	332
10.33.11 imap_fetchstructure [Notes en ligne] [Exemples]	332
10.33.12 imap_headerinfo [Notes en ligne] [Exemples]	334
10.33.13 imap_header [Notes en ligne] [Exemples]	335
10.33.14 imap_rfc822_parse_headers [Notes en ligne] [Exemples]	335
10.33.15 imap_headers [Notes en ligne] [Exemples]	335
10.33.16 imap_listmailbox [Notes en ligne] [Exemples]	335
10.33.17 imap_getmailboxes [Notes en ligne] [Exemples]	336
10.33.18 imap_listsubscribed [Notes en ligne] [Exemples]	337
10.33.19 imap_getsubscribed [Notes en ligne] [Exemples]	337
10.33.20 imap_mail_copy [Notes en ligne] [Exemples]	337
10.33.21 imap_mail_move [Notes en ligne] [Exemples]	338
10.33.22 imap_num_msg [Notes en ligne] [Exemples]	338
10.33.23 imap_num_recent [Notes en ligne] [Exemples]	338
10.33.24 imap_open [Notes en ligne] [Exemples]	338
10.33.25 imap_ping [Notes en ligne] [Exemples]	339
10.33.26 imap_renamemailbox [Notes en ligne] [Exemples]	340
10.33.27 imap_reopen [Notes en ligne] [Exemples]	340
10.33.28 imap_subscribe [Notes en ligne] [Exemples]	340
10.33.29 imap_undelete [Notes en ligne] [Exemples]	341
10.33.30 imap_unsubscribe [Notes en ligne] [Exemples]	341
10.33.31 imap_qprint [Notes en ligne] [Exemples]	341
10.33.32 imap_8bit [Notes en ligne] [Exemples]	341
10.33.33 imap_binary [Notes en ligne] [Exemples]	341
10.33.34 imap_scanmailbox [Notes en ligne] [Exemples]	342
10.33.35 imap_mailboxmsginfo [Notes en ligne] [Exemples]	342
10.33.36 imap_rfc822_write_address [Notes en ligne] [Exemples]	343
10.33.37 imap_rfc822_parse_adrlist [Notes en ligne] [Exemples]	343
10.33.38 imap_setflag_full [Notes en ligne] [Exemples]	344
10.33.39 imap_clearflag_full [Notes en ligne] [Exemples]	344
10.33.40 imap_sort [Notes en ligne] [Exemples]	345
10.33.41 imap_fetchheader [Notes en ligne] [Exemples]	345
10.33.42 imap_uid [Notes en ligne] [Exemples]	345
10.33.43 imap_msgno [Notes en ligne] [Exemples]	346
10.33.44 imap_search [Notes en ligne] [Exemples]	346
10.33.45 imap_last_error [Notes en ligne] [Exemples]	347
10.33.46 imap_errors [Notes en ligne] [Exemples]	347
10.33.47 imap_alerts [Notes en ligne] [Exemples]	347
10.33.48 imap_status [Notes en ligne] [Exemples]	347
10.33.49 imap_utf7_decode [Notes en ligne] [Exemples]	348
10.33.50 imap_utf7_encode [Notes en ligne] [Exemples]	348
10.33.51 imap_utf8 [Notes en ligne] [Exemples]	349
10.33.52 imap_fetch_overview [Notes en ligne] [Exemples]	349
10.33.53 imap_mime_header_decode [Notes en ligne] [Exemples]	350
10.33.54 imap_mail_compose [Notes en ligne] [Exemples]	350

Table of Contents

10.33.55	imap_mail [Notes en ligne] [Exemples]	351
10.34	Options PHP & informations [Notes en ligne]	351
10.34.1	assert [Notes en ligne] [Exemples]	351
10.34.2	assert_options [Notes en ligne] [Exemples]	352
10.34.3	extension_loaded [Notes en ligne] [Exemples]	352
10.34.4	dl [Notes en ligne] [Exemples]	352
10.34.5	getenv [Notes en ligne] [Exemples]	353
10.34.6	get_cfg_var [Notes en ligne] [Exemples]	353
10.34.7	get_current_user [Notes en ligne] [Exemples]	353
10.34.8	get_magic_quotes_gpc [Notes en ligne] [Exemples]	353
10.34.9	get_magic_quotes_runtime [Notes en ligne] [Exemples]	354
10.34.10	getlastmod [Notes en ligne] [Exemples]	354
10.34.11	getmyinode [Notes en ligne] [Exemples]	354
10.34.12	getmypid [Notes en ligne] [Exemples]	354
10.34.13	getmyuid [Notes en ligne] [Exemples]	355
10.34.14	getrusage [Notes en ligne] [Exemples]	355
10.34.15	ini_alter [Notes en ligne] [Exemples]	355
10.34.16	ini_get [Notes en ligne] [Exemples]	356
10.34.17	ini_restore [Notes en ligne] [Exemples]	356
10.34.18	ini_set [Notes en ligne] [Exemples]	356
10.34.19	phpcredits [Notes en ligne] [Exemples]	356
10.34.20	phpinfo [Notes en ligne] [Exemples]	358
10.34.21	phpversion [Notes en ligne] [Exemples]	358
10.34.22	php_logo_guid [Notes en ligne] [Exemples]	358
10.34.23	php_sapi_name [Notes en ligne] [Exemples]	359
10.34.24	php_uname [Notes en ligne] [Exemples]	359
10.34.25	putenv [Notes en ligne] [Exemples]	359
10.34.26	set_magic_quotes_runtime [Notes en ligne] [Exemples]	360
10.34.27	set_time_limit [Notes en ligne] [Exemples]	360
10.34.28	zend_logo_guid [Notes en ligne] [Exemples]	360
10.34.29	get_loaded_extensions [Notes en ligne] [Exemples]	360
10.34.30	get_extension_funcs [Notes en ligne] [Exemples]	361
10.34.31	get_required_files [Notes en ligne] [Exemples]	361
10.34.32	get_included_files [Notes en ligne] [Exemples]	362
10.35	Ingres II [Notes en ligne]	362
10.35.1	ingres_connect [Notes en ligne] [Exemples]	362
10.35.2	ingres_pconnect [Notes en ligne] [Exemples]	363
10.35.3	ingres_close [Notes en ligne] [Exemples]	363
10.35.4	ingres_query [Notes en ligne] [Exemples]	364
10.35.5	ingres_num_rows [Notes en ligne] [Exemples]	365
10.35.6	ingres_num_fields [Notes en ligne] [Exemples]	365
10.35.7	ingres_field_name [Notes en ligne] [Exemples]	365
10.35.8	ingres_field_type [Notes en ligne] [Exemples]	365
10.35.9	ingres_field_nullable [Notes en ligne] [Exemples]	366
10.35.10	ingres_field_length [Notes en ligne] [Exemples]	366
10.35.11	ingres_field_precision [Notes en ligne] [Exemples]	366
10.35.12	ingres_field_scale [Notes en ligne] [Exemples]	366

Table of Contents

10.35.13	ingres_fetch_array [Notes en ligne] [Exemples]	367
10.35.14	ingres_fetch_row [Notes en ligne] [Exemples]	367
10.35.15	ingres_fetch_object [Notes en ligne] [Exemples]	368
10.35.16	ingres_rollback [Notes en ligne] [Exemples]	368
10.35.17	ingres_commit [Notes en ligne] [Exemples]	369
10.35.18	ingres_autocommit [Notes en ligne] [Exemples]	369
10.36	LDAP [Notes en ligne]	369
10.36.1	Introduction à LDAP [Notes en ligne]	369
10.36.2	Exemple complet [Notes en ligne]	370
10.36.2.1	Utilisation des fonctions PHP LDAP [Notes en ligne]	371
10.36.2.2	Plus d'informations [Notes en ligne]	371
10.36.3	ldap_add [Notes en ligne] [Exemples]	371
10.36.4	ldap_bind [Notes en ligne] [Exemples]	372
10.36.5	ldap_close [Notes en ligne] [Exemples]	372
10.36.6	ldap_compare [Notes en ligne] [Exemples]	373
10.36.7	ldap_connect [Notes en ligne] [Exemples]	373
10.36.8	ldap_count_entries [Notes en ligne] [Exemples]	374
10.36.9	ldap_delete [Notes en ligne] [Exemples]	374
10.36.10	ldap_dn2ufn [Notes en ligne] [Exemples]	374
10.36.11	ldap_err2str [Notes en ligne] [Exemples]	374
10.36.12	ldap_errno [Notes en ligne] [Exemples]	375
10.36.13	ldap_error [Notes en ligne] [Exemples]	375
10.36.14	ldap_explode_dn [Notes en ligne] [Exemples]	376
10.36.15	ldap_first_attribute [Notes en ligne] [Exemples]	376
10.36.16	ldap_first_entry [Notes en ligne] [Exemples]	376
10.36.17	ldap_free_result [Notes en ligne] [Exemples]	377
10.36.18	ldap_get_attributes [Notes en ligne] [Exemples]	377
10.36.19	ldap_get_dn [Notes en ligne] [Exemples]	377
10.36.20	ldap_get_entries [Notes en ligne] [Exemples]	378
10.36.21	ldap_get_option [Notes en ligne] [Exemples]	378
10.36.22	ldap_get_values [Notes en ligne] [Exemples]	379
10.36.23	ldap_get_values_len [Notes en ligne] [Exemples]	379
10.36.24	ldap_list [Notes en ligne] [Exemples]	380
10.36.25	ldap_modify [Notes en ligne] [Exemples]	380
10.36.26	ldap_mod_add [Notes en ligne] [Exemples]	380
10.36.27	ldap_mod_del [Notes en ligne] [Exemples]	381
10.36.28	ldap_mod_replace [Notes en ligne] [Exemples]	381
10.36.29	ldap_next_attribute [Notes en ligne] [Exemples]	381
10.36.30	ldap_next_entry [Notes en ligne] [Exemples]	381
10.36.31	ldap_read [Notes en ligne] [Exemples]	382
10.36.32	ldap_search [Notes en ligne] [Exemples]	382
10.36.33	ldap_set_option [Notes en ligne] [Exemples]	383
10.36.34	ldap_unbind [Notes en ligne] [Exemples]	384
10.37	Email [Notes en ligne]	384
10.37.1	mail [Notes en ligne] [Exemples]	384
10.37.2	ezmlm_hash [Notes en ligne] [Exemples]	385
10.38	Mathématiques [Notes en ligne]	386

Table of Contents

10.38.1	Introduction [Notes en ligne]	386
10.38.1.1	Constantes mathématiques [Notes en ligne]	386
10.38.2	abs [Notes en ligne] [Exemples]	387
10.38.3	acos [Notes en ligne] [Exemples]	387
10.38.4	asin [Notes en ligne] [Exemples]	387
10.38.5	atan [Notes en ligne] [Exemples]	387
10.38.6	atan2 [Notes en ligne] [Exemples]	387
10.38.7	base_convert [Notes en ligne] [Exemples]	388
10.38.8	bindec [Notes en ligne] [Exemples]	388
10.38.9	ceil [Notes en ligne] [Exemples]	388
10.38.10	cos [Notes en ligne] [Exemples]	389
10.38.11	decbin [Notes en ligne] [Exemples]	389
10.38.12	dechex [Notes en ligne] [Exemples]	389
10.38.13	decoct [Notes en ligne] [Exemples]	389
10.38.14	deg2rad [Notes en ligne] [Exemples]	389
10.38.15	exp [Notes en ligne] [Exemples]	390
10.38.16	floor [Notes en ligne] [Exemples]	390
10.38.17	getrandmax [Notes en ligne] [Exemples]	390
10.38.18	hexdec [Notes en ligne] [Exemples]	390
10.38.19	lcg_value [Notes en ligne] [Exemples]	390
10.38.20	log [Notes en ligne] [Exemples]	391
10.38.21	log10 [Notes en ligne] [Exemples]	391
10.38.22	max [Notes en ligne] [Exemples]	391
10.38.23	min [Notes en ligne] [Exemples]	391
10.38.24	mt_rand [Notes en ligne] [Exemples]	392
10.38.25	mt_srand [Notes en ligne] [Exemples]	392
10.38.26	mt_getrandmax [Notes en ligne] [Exemples]	392
10.38.27	number_format [Notes en ligne] [Exemples]	393
10.38.28	octdec [Notes en ligne] [Exemples]	393
10.38.29	pi [Notes en ligne] [Exemples]	393
10.38.30	pow [Notes en ligne] [Exemples]	393
10.38.31	rad2deg [Notes en ligne] [Exemples]	393
10.38.32	rand [Notes en ligne] [Exemples]	394
10.38.33	round [Notes en ligne] [Exemples]	394
10.38.34	sin [Notes en ligne] [Exemples]	394
10.38.35	sqrt [Notes en ligne] [Exemples]	394
10.38.36	srand [Notes en ligne] [Exemples]	395
10.38.37	tan [Notes en ligne] [Exemples]	395
10.39	MCAL [Notes en ligne]	395
10.39.1	mcal_open [Notes en ligne] [Exemples]	396
10.39.2	mcal_popen [Notes en ligne] [Exemples]	396
10.39.3	mcal_reopen [Notes en ligne] [Exemples]	396
10.39.4	mcal_close [Notes en ligne] [Exemples]	396
10.39.5	mcal_create_calendar [Notes en ligne] [Exemples]	397
10.39.6	mcal_rename_calendar [Notes en ligne] [Exemples]	397
10.39.7	mcal_delete_calendar [Notes en ligne] [Exemples]	397
10.39.8	mcal_fetch_event [Notes en ligne] [Exemples]	397

Table of Contents

10.39.9 mcal_list_events [Notes en ligne] [Exemples]	398
10.39.10 mcal_append_event [Notes en ligne] [Exemples]	398
10.39.11 mcal_store_event [Notes en ligne] [Exemples]	398
10.39.12 mcal_delete_event [Notes en ligne] [Exemples]	398
10.39.13 mcal_snooze [Notes en ligne] [Exemples]	399
10.39.14 mcal_list_alarms [Notes en ligne] [Exemples]	399
10.39.15 mcal_event_init [Notes en ligne] [Exemples]	399
10.39.16 mcal_event_set_category [Notes en ligne] [Exemples]	399
10.39.17 mcal_event_set_title [Notes en ligne] [Exemples]	399
10.39.18 mcal_event_set_description [Notes en ligne] [Exemples]	400
10.39.19 mcal_event_set_start [Notes en ligne] [Exemples]	400
10.39.20 mcal_event_set_end [Notes en ligne] [Exemples]	400
10.39.21 mcal_event_set_alarm [Notes en ligne] [Exemples]	400
10.39.22 mcal_event_set_class [Notes en ligne] [Exemples]	400
10.39.23 mcal_is_leap_year [Notes en ligne] [Exemples]	401
10.39.24 mcal_days_in_month [Notes en ligne] [Exemples]	401
10.39.25 mcal_date_valid [Notes en ligne] [Exemples]	401
10.39.26 mcal_time_valid [Notes en ligne] [Exemples]	401
10.39.27 mcal_day_of_week [Notes en ligne] [Exemples]	401
10.39.28 mcal_day_of_year [Notes en ligne] [Exemples]	402
10.39.29 mcal_date_compare [Notes en ligne] [Exemples]	402
10.39.30 mcal_next_recurrence [Notes en ligne] [Exemples]	402
10.39.31 mcal_event_set_recur_none [Notes en ligne] [Exemples]	402
10.39.32 mcal_event_set_recur_daily [Notes en ligne] [Exemples]	402
10.39.33 mcal_event_set_recur_weekly [Notes en ligne] [Exemples]	403
10.39.34 mcal_event_set_recur_monthly_mday [Notes en ligne] [Exemples]	403
10.39.35 mcal_event_set_recur_monthly_wday [Notes en ligne] [Exemples]	403
10.39.36 mcal_event_set_recur_yearly [Notes en ligne] [Exemples]	403
10.39.37 mcal_fetch_current_stream_event [Notes en ligne] [Exemples]	403
10.39.38 mcal_event_add_attribute [Notes en ligne] [Exemples]	404
10.39.39 mcal_expunge [Notes en ligne] [Exemples]	404
10.40 Cryptage [Notes en ligne]	404
10.40.1 mcrypt_get_cipher_name [Notes en ligne] [Exemples]	407
10.40.2 mcrypt_get_block_size [Notes en ligne] [Exemples]	407
10.40.3 mcrypt_get_key_size [Notes en ligne] [Exemples]	407
10.40.4 mcrypt_create_iv [Notes en ligne] [Exemples]	408
10.40.5 mcrypt_cbc [Notes en ligne] [Exemples]	408
10.40.6 mcrypt_cfb [Notes en ligne] [Exemples]	409
10.40.7 mcrypt_ecb [Notes en ligne] [Exemples]	409
10.40.8 mcrypt_ofb [Notes en ligne] [Exemples]	409
10.40.9 mcrypt_list_algorithms [Notes en ligne] [Exemples]	410
10.40.10 mcrypt_list_modes [Notes en ligne] [Exemples]	410
10.40.11 mcrypt_get_iv_size [Notes en ligne] [Exemples]	411
10.40.12 mcrypt_encrypt [Notes en ligne] [Exemples]	411
10.40.13 mcrypt_decrypt [Notes en ligne] [Exemples]	412
10.40.14 mcrypt_module_open [Notes en ligne] [Exemples]	412
10.40.15 mcrypt_generic_init [Notes en ligne] [Exemples]	413

Table of Contents

10.40.16	mdecrypt_generic [Notes en ligne] [Exemples]	413
10.40.17	mdecrypt_generic [Notes en ligne] [Exemples]	413
10.40.18	mdecrypt_generic_end [Notes en ligne] [Exemples]	414
10.40.19	mdecrypt_enc_self_test [Notes en ligne] [Exemples]	414
10.40.20	mdecrypt_enc_is_block_algorithm_mode [Notes en ligne] [Exemples]	414
10.40.21	mdecrypt_enc_is_block_algorithm [Notes en ligne] [Exemples]	414
10.40.22	mdecrypt_enc_is_block_mode [Notes en ligne] [Exemples]	414
10.40.23	mdecrypt_enc_get_block_size [Notes en ligne] [Exemples]	415
10.40.24	mdecrypt_enc_get_key_size [Notes en ligne] [Exemples]	415
10.40.25	mdecrypt_enc_get_supported_key_sizes [Notes en ligne] [Exemples]	415
10.40.26	mdecrypt_enc_get_iv_size [Notes en ligne] [Exemples]	415
10.40.27	mdecrypt_enc_get_algorithms_name [Notes en ligne] [Exemples]	415
10.40.28	mdecrypt_enc_get_modes_name [Notes en ligne] [Exemples]	416
10.40.29	mdecrypt_module_self_test [Notes en ligne] [Exemples]	416
10.40.30	mdecrypt_module_is_block_algorithm_mode [Notes en ligne] [Exemples]	416
10.40.31	mdecrypt_module_is_block_algorithm [Notes en ligne] [Exemples]	416
10.40.32	mdecrypt_module_is_block_mode [Notes en ligne] [Exemples]	416
10.40.33	mdecrypt_module_get_algo_block_size [Notes en ligne] [Exemples]	417
10.40.34	mdecrypt_module_get_algo_key_size [Notes en ligne] [Exemples]	417
10.40.35	mdecrypt_module_get_algo_supported_key_sizes [Notes en ligne] [Exemples]	417
10.41	Hash [Notes en ligne]	417
10.41.1	mhash_get_hash_name [Notes en ligne] [Exemples]	418
10.41.2	mhash_get_block_size [Notes en ligne] [Exemples]	418
10.41.3	mhash_count [Notes en ligne] [Exemples]	419
10.41.4	mhash [Notes en ligne] [Exemples]	419
10.41.5	mhash_keygen_s2k [Notes en ligne] [Exemples]	419
10.42	Fonctions diverses [Notes en ligne]	419
10.42.1	connection_aborted [Notes en ligne] [Exemples]	420
10.42.2	connection_status [Notes en ligne] [Exemples]	420
10.42.3	connection_timeout [Notes en ligne] [Exemples]	420
10.42.4	define [Notes en ligne] [Exemples]	420
10.42.5	defined [Notes en ligne] [Exemples]	421
10.42.6	die [Notes en ligne] [Exemples]	421
10.42.7	eval [Notes en ligne] [Exemples]	421
10.42.8	exit [Notes en ligne] [Exemples]	422
10.42.9	get_browser [Notes en ligne] [Exemples]	422
10.42.10	highlight_file [Notes en ligne] [Exemples]	423
10.42.11	highlight_string [Notes en ligne] [Exemples]	424
10.42.12	ignore_user_abort [Notes en ligne] [Exemples]	424
10.42.13	iptcp_parse [Notes en ligne] [Exemples]	424
10.42.14	leak [Notes en ligne] [Exemples]	425
10.42.15	pack [Notes en ligne] [Exemples]	425
10.42.16	show_source [Notes en ligne] [Exemples]	426
10.42.17	sleep [Notes en ligne] [Exemples]	426
10.42.18	uniqid [Notes en ligne] [Exemples]	426
10.42.19	unpack [Notes en ligne] [Exemples]	427
10.42.20	usleep [Notes en ligne] [Exemples]	427

Table of Contents

10.43 mSQL [Notes en ligne]	427
10.43.1 msql [Notes en ligne] [Exemples]	428
10.43.2 msql affected rows [Notes en ligne] [Exemples]	428
10.43.3 msql close [Notes en ligne] [Exemples]	428
10.43.4 msql connect [Notes en ligne] [Exemples]	428
10.43.5 msql create_db [Notes en ligne] [Exemples]	429
10.43.6 msql createdb [Notes en ligne] [Exemples]	429
10.43.7 msql data_seek [Notes en ligne] [Exemples]	429
10.43.8 msql dbname [Notes en ligne] [Exemples]	429
10.43.9 msql drop_db [Notes en ligne] [Exemples]	429
10.43.10 msql dropdb [Notes en ligne] [Exemples]	430
10.43.11 msql error [Notes en ligne] [Exemples]	430
10.43.12 msql fetch_array [Notes en ligne] [Exemples]	430
10.43.13 msql fetch_field [Notes en ligne] [Exemples]	430
10.43.14 msql fetch_object [Notes en ligne] [Exemples]	431
10.43.15 msql fetch_row [Notes en ligne] [Exemples]	431
10.43.16 msql fieldname [Notes en ligne] [Exemples]	431
10.43.17 msql field_seek [Notes en ligne] [Exemples]	432
10.43.18 msql fieldtable [Notes en ligne] [Exemples]	432
10.43.19 msql fieldtype [Notes en ligne] [Exemples]	432
10.43.20 msql fieldflags [Notes en ligne] [Exemples]	432
10.43.21 msql fieldlen [Notes en ligne] [Exemples]	432
10.43.22 msql free_result [Notes en ligne] [Exemples]	433
10.43.23 msql freeresult [Notes en ligne] [Exemples]	433
10.43.24 msql list_fields [Notes en ligne] [Exemples]	433
10.43.25 msql listfields [Notes en ligne] [Exemples]	433
10.43.26 msql list_dbs [Notes en ligne] [Exemples]	433
10.43.27 msql listdbs [Notes en ligne] [Exemples]	434
10.43.28 msql list_tables [Notes en ligne] [Exemples]	434
10.43.29 msql listtables [Notes en ligne] [Exemples]	434
10.43.30 msql num_fields [Notes en ligne] [Exemples]	434
10.43.31 msql num_rows [Notes en ligne] [Exemples]	434
10.43.32 msql numfields [Notes en ligne] [Exemples]	434
10.43.33 msql numrows [Notes en ligne] [Exemples]	435
10.43.34 msql pconnect [Notes en ligne] [Exemples]	435
10.43.35 msql query [Notes en ligne] [Exemples]	435
10.43.36 msql regcase [Notes en ligne] [Exemples]	435
10.43.37 msql result [Notes en ligne] [Exemples]	435
10.43.38 msql select_db [Notes en ligne] [Exemples]	436
10.43.39 msql selectdb [Notes en ligne] [Exemples]	436
10.43.40 msql tablename [Notes en ligne] [Exemples]	436
10.44 Microsoft SQL Server [Notes en ligne]	437
10.44.1 mssql_close [Notes en ligne] [Exemples]	437
10.44.2 mssql_connect [Notes en ligne] [Exemples]	437
10.44.3 mssql_data_seek [Notes en ligne] [Exemples]	437
10.44.4 mssql_fetch_array [Notes en ligne] [Exemples]	438
10.44.5 mssql_fetch_field [Notes en ligne] [Exemples]	438

Table of Contents

10.44.6 mssql_fetch_object [Notes en ligne] [Exemples]	438
10.44.7 mssql_fetch_row [Notes en ligne] [Exemples]	439
10.44.8 mssql_field_length [Notes en ligne] [Exemples]	439
10.44.9 mssql_field_name [Notes en ligne] [Exemples]	439
10.44.10 mssql_field_seek [Notes en ligne] [Exemples]	439
10.44.11 mssql_field_type [Notes en ligne] [Exemples]	439
10.44.12 mssql_free_result [Notes en ligne] [Exemples]	439
10.44.13 mssql_get_last_message [Notes en ligne] [Exemples]	440
10.44.14 mssql_min_error_severity [Notes en ligne] [Exemples]	440
10.44.15 mssql_min_message_severity [Notes en ligne] [Exemples]	440
10.44.16 mssql_num_fields [Notes en ligne] [Exemples]	440
10.44.17 mssql_num_rows [Notes en ligne] [Exemples]	440
10.44.18 mssql_pconnect [Notes en ligne] [Exemples]	440
10.44.19 mssql_query [Notes en ligne] [Exemples]	441
10.44.20 mssql_result [Notes en ligne] [Exemples]	441
10.44.21 mssql_select_db [Notes en ligne] [Exemples]	441
10.45 MySQL [Notes en ligne]	442
10.45.1 mysql_affected_rows [Notes en ligne] [Exemples]	442
10.45.2 mysql_change_user [Notes en ligne] [Exemples]	442
10.45.3 mysql_close [Notes en ligne] [Exemples]	443
10.45.4 mysql_connect [Notes en ligne] [Exemples]	443
10.45.5 mysql_create_db [Notes en ligne] [Exemples]	444
10.45.6 mysql_data_seek [Notes en ligne] [Exemples]	444
10.45.7 mysql_db_name [Notes en ligne] [Exemples]	445
10.45.8 mysql_db_query [Notes en ligne] [Exemples]	445
10.45.9 mysql_drop_db [Notes en ligne] [Exemples]	446
10.45.10 mysql_errno [Notes en ligne] [Exemples]	446
10.45.11 mysql_error [Notes en ligne] [Exemples]	446
10.45.12 mysql_fetch_array [Notes en ligne] [Exemples]	447
10.45.13 mysql_fetch_assoc [Notes en ligne] [Exemples]	447
10.45.14 mysql_fetch_field [Notes en ligne] [Exemples]	448
10.45.15 mysql_fetch_lengths [Notes en ligne] [Exemples]	448
10.45.16 mysql_fetch_object [Notes en ligne] [Exemples]	449
10.45.17 mysql_fetch_row [Notes en ligne] [Exemples]	449
10.45.18 mysql_field_flags [Notes en ligne] [Exemples]	450
10.45.19 mysql_field_name [Notes en ligne] [Exemples]	450
10.45.20 mysql_field_len [Notes en ligne] [Exemples]	450
10.45.21 mysql_field_seek [Notes en ligne] [Exemples]	450
10.45.22 mysql_field_table [Notes en ligne] [Exemples]	451
10.45.23 mysql_field_type [Notes en ligne] [Exemples]	451
10.45.24 mysql_free_result [Notes en ligne] [Exemples]	451
10.45.25 mysql_insert_id [Notes en ligne] [Exemples]	452
10.45.26 mysql_list_dbs [Notes en ligne] [Exemples]	452
10.45.27 mysql_list_fields [Notes en ligne] [Exemples]	452
10.45.28 mysql_list_tables [Notes en ligne] [Exemples]	452
10.45.29 mysql_num_fields [Notes en ligne] [Exemples]	453
10.45.30 mysql_num_rows [Notes en ligne] [Exemples]	453

Table of Contents

10.45.31	mysql_pconnect [Notes en ligne] [Exemples]	453
10.45.32	mysql_query [Notes en ligne] [Exemples]	454
10.45.33	mysql_result [Notes en ligne] [Exemples]	455
10.45.34	mysql_select_db [Notes en ligne] [Exemples]	455
10.45.35	mysql_tablename [Notes en ligne] [Exemples]	455
10.46	Réseau [Notes en ligne]	456
10.46.1	checkdnsrr [Notes en ligne] [Exemples]	456
10.46.2	closelog [Notes en ligne] [Exemples]	456
10.46.3	debugger_off [Notes en ligne] [Exemples]	456
10.46.4	debugger_on [Notes en ligne] [Exemples]	456
10.46.5	define_syslog_variables [Notes en ligne] [Exemples]	457
10.46.6	fsockopen [Notes en ligne] [Exemples]	457
10.46.7	gethostbyaddr [Notes en ligne] [Exemples]	458
10.46.8	gethostbyname [Notes en ligne] [Exemples]	458
10.46.9	gethostbyname1 [Notes en ligne] [Exemples]	458
10.46.10	getmxrr [Notes en ligne] [Exemples]	458
10.46.11	getprotobyname [Notes en ligne] [Exemples]	459
10.46.12	getprotobynumber [Notes en ligne] [Exemples]	459
10.46.13	getservbyname [Notes en ligne] [Exemples]	459
10.46.14	getservbyport [Notes en ligne] [Exemples]	459
10.46.15	ip2long [Notes en ligne] [Exemples]	459
10.46.16	long2ip [Notes en ligne] [Exemples]	460
10.46.17	openlog [Notes en ligne] [Exemples]	460
10.46.18	pfsockopen [Notes en ligne] [Exemples]	461
10.46.19	socket_get_status [Notes en ligne] [Exemples]	461
10.46.20	socket_set_blocking [Notes en ligne] [Exemples]	462
10.46.21	socket_set_timeout [Notes en ligne] [Exemples]	462
10.46.22	syslog [Notes en ligne] [Exemples]	462
10.47	NIS [Notes en ligne]	463
10.47.1	yp_get_default_domain [Notes en ligne] [Exemples]	464
10.47.2	yp_order [Notes en ligne] [Exemples]	464
10.47.3	yp_master [Notes en ligne] [Exemples]	464
10.47.4	yp_match [Notes en ligne] [Exemples]	465
10.47.5	yp_first [Notes en ligne] [Exemples]	465
10.47.6	yp_next [Notes en ligne] [Exemples]	466
10.48	Oracle 8 [Notes en ligne]	466
10.48.1	ocidefinebyname [Notes en ligne] [Exemples]	467
10.48.2	ocibindbyname [Notes en ligne] [Exemples]	468
10.48.3	ocilogon [Notes en ligne] [Exemples]	469
10.48.4	ociploton [Notes en ligne] [Exemples]	470
10.48.5	ocinlogon [Notes en ligne] [Exemples]	470
10.48.6	ocilogoff [Notes en ligne] [Exemples]	472
10.48.7	ociexecute [Notes en ligne] [Exemples]	472
10.48.8	ocicommit [Notes en ligne] [Exemples]	472
10.48.9	ocirollback [Notes en ligne] [Exemples]	472
10.48.10	ocinewdescriptor [Notes en ligne] [Exemples]	472
10.48.11	ocirowcount [Notes en ligne] [Exemples]	474

Table of Contents

10.48.12	ocinumcols [Notes en ligne] [Exemples]	474
10.48.13	ocireult [Notes en ligne] [Exemples]	475
10.48.14	ocifetch [Notes en ligne] [Exemples]	475
10.48.15	ocifetchinto [Notes en ligne] [Exemples]	475
10.48.16	ocifetchstatement [Notes en ligne] [Exemples]	476
10.48.17	ocicolumnisnull [Notes en ligne] [Exemples]	476
10.48.18	ocicolumnname [Notes en ligne] [Exemples]	476
10.48.19	ocicolumnsize [Notes en ligne] [Exemples]	477
10.48.20	ocicolumntype [Notes en ligne] [Exemples]	478
10.48.21	ociserverversion [Notes en ligne] [Exemples]	479
10.48.22	ocistatementtype [Notes en ligne] [Exemples]	479
10.48.23	ocinewcursor [Notes en ligne] [Exemples]	480
10.48.24	ocifreestatement [Notes en ligne] [Exemples]	481
10.48.25	ocifreecursor [Notes en ligne] [Exemples]	481
10.48.26	ocifreedesc [Notes en ligne] [Exemples]	481
10.48.27	ociparse [Notes en ligne] [Exemples]	481
10.48.28	ocierror [Notes en ligne] [Exemples]	482
10.48.29	ociinternaldebug [Notes en ligne] [Exemples]	482
10.49	OpenSSL [Notes en ligne]	482
10.49.1	openssl_free_key [Notes en ligne] [Exemples]	482
10.49.2	openssl_get_privatekey [Notes en ligne] [Exemples]	482
10.49.3	openssl_get_publickey [Notes en ligne] [Exemples]	483
10.49.4	openssl_open [Notes en ligne] [Exemples]	483
10.49.5	openssl_seal [Notes en ligne] [Exemples]	483
10.49.6	openssl_sign [Notes en ligne] [Exemples]	484
10.49.7	openssl_verify [Notes en ligne] [Exemples]	485
10.50	Oracle [Notes en ligne]	485
10.50.1	ora_bind [Notes en ligne] [Exemples]	485
10.50.2	ora_close [Notes en ligne] [Exemples]	486
10.50.3	ora_columnname [Notes en ligne] [Exemples]	486
10.50.4	ora_columnsize [Notes en ligne] [Exemples]	486
10.50.5	ora_columntype [Notes en ligne] [Exemples]	486
10.50.6	ora_commit [Notes en ligne] [Exemples]	487
10.50.7	ora_commitoff [Notes en ligne] [Exemples]	487
10.50.8	ora_commiton [Notes en ligne] [Exemples]	487
10.50.9	ora_do [Notes en ligne] [Exemples]	488
10.50.10	ora_error [Notes en ligne] [Exemples]	488
10.50.11	ora_errorcode [Notes en ligne] [Exemples]	488
10.50.12	ora_exec [Notes en ligne] [Exemples]	488
10.50.13	ora_fetch [Notes en ligne] [Exemples]	489
10.50.14	ora_fetch_into [Notes en ligne] [Exemples]	489
10.50.15	ora_getcolumn [Notes en ligne] [Exemples]	489
10.50.16	ora_logoff [Notes en ligne] [Exemples]	489
10.50.17	ora_logon [Notes en ligne] [Exemples]	490
10.50.18	ora_plogon [Notes en ligne] [Exemples]	490
10.50.19	ora_numcols [Notes en ligne] [Exemples]	490
10.50.20	ora_numrows [Notes en ligne] [Exemples]	490

Table of Contents

10.50.21	ora_open [Notes en ligne] [Exemples]	491
10.50.22	ora_parse [Notes en ligne] [Exemples]	491
10.50.23	ora_rollback [Notes en ligne] [Exemples]	491
10.51	Entrées/sorties [Notes en ligne]	491
10.51.1	flush [Notes en ligne] [Exemples]	492
10.51.2	ob_start [Notes en ligne] [Exemples]	492
10.51.3	ob_get_contents [Notes en ligne] [Exemples]	493
10.51.4	ob_get_length [Notes en ligne] [Exemples]	493
10.51.5	ob_end_flush [Notes en ligne] [Exemples]	493
10.51.6	ob_end_clean [Notes en ligne] [Exemples]	493
10.51.7	ob_implicit_flush [Notes en ligne] [Exemples]	494
10.52	Ovrimos SQL [Notes en ligne]	494
10.52.1	ovrimos_connect [Notes en ligne] [Exemples]	494
10.52.2	ovrimos_close [Notes en ligne] [Exemples]	495
10.52.3	ovrimos_close_all [Notes en ligne] [Exemples]	495
10.52.4	ovrimos_longreadlen [Notes en ligne] [Exemples]	495
10.52.5	ovrimos_prepare [Notes en ligne] [Exemples]	496
10.52.6	ovrimos_execute [Notes en ligne] [Exemples]	496
10.52.7	ovrimos_cursor [Notes en ligne] [Exemples]	496
10.52.8	ovrimos_exec [Notes en ligne] [Exemples]	497
10.52.9	ovrimos_fetch_into [Notes en ligne] [Exemples]	497
10.52.10	ovrimos_fetch_row [Notes en ligne] [Exemples]	498
10.52.11	ovrimos_result [Notes en ligne] [Exemples]	498
10.52.12	ovrimos_result_all [Notes en ligne] [Exemples]	499
10.52.13	ovrimos_num_rows [Notes en ligne] [Exemples]	500
10.52.14	ovrimos_num_fields [Notes en ligne] [Exemples]	500
10.52.15	ovrimos_field_name [Notes en ligne] [Exemples]	500
10.52.16	ovrimos_field_type [Notes en ligne] [Exemples]	501
10.52.17	ovrimos_field_len [Notes en ligne] [Exemples]	501
10.52.18	ovrimos_field_num [Notes en ligne] [Exemples]	501
10.52.19	ovrimos_free_result [Notes en ligne] [Exemples]	501
10.52.20	ovrimos_commit [Notes en ligne] [Exemples]	501
10.52.21	ovrimos_rollback [Notes en ligne] [Exemples]	502
10.53	Expressions régulières compatibles Perl [Notes en ligne]	502
10.53.1	preg_match [Notes en ligne] [Exemples]	502
10.53.2	preg_match_all [Notes en ligne] [Exemples]	503
10.53.3	preg_replace [Notes en ligne] [Exemples]	505
10.53.4	preg_split [Notes en ligne] [Exemples]	506
10.53.5	preg_quote [Notes en ligne] [Exemples]	507
10.53.6	preg_grep [Notes en ligne] [Exemples]	508
10.53.7	options de recherche [Notes en ligne]	508
10.53.8	syntaxe des masques [Notes en ligne]	509
10.54	PDF [Notes en ligne]	523
10.54.1	Confusion entre les vieilles version de pdflib [Notes en ligne]	523
10.54.2	Conseils pour installer pdflib 3.x [Notes en ligne]	525
10.54.3	Installation des anciennes versions de pdflib [Notes en ligne]	525
10.54.4	Exemples [Notes en ligne]	525

Table of Contents

10.54.5 pdf_set_info [Notes en ligne] [Exemples]	527
10.54.6 pdf_open [Notes en ligne] [Exemples]	528
10.54.7 pdf_close [Notes en ligne] [Exemples]	528
10.54.8 pdf_begin_page [Notes en ligne] [Exemples]	528
10.54.9 pdf_end_page [Notes en ligne] [Exemples]	529
10.54.10 pdf_show [Notes en ligne] [Exemples]	529
10.54.11 pdf_show_boxed [Notes en ligne] [Exemples]	529
10.54.12 pdf_show_xy [Notes en ligne] [Exemples]	529
10.54.13 pdf_set_font [Notes en ligne] [Exemples]	530
10.54.14 pdf_set_leading [Notes en ligne] [Exemples]	530
10.54.15 pdf_set_parameter [Notes en ligne] [Exemples]	530
10.54.16 pdf_get_parameter [Notes en ligne] [Exemples]	530
10.54.17 pdf_set_value [Notes en ligne] [Exemples]	531
10.54.18 pdf_get_value [Notes en ligne] [Exemples]	531
10.54.19 pdf_get_image_height [Notes en ligne] [Exemples]	531
10.54.20 pdf_get_image_width [Notes en ligne] [Exemples]	531
10.54.21 pdf_set_text_rendering [Notes en ligne] [Exemples]	531
10.54.22 pdf_set_horiz_scaling [Notes en ligne] [Exemples]	532
10.54.23 pdf_set_text_rise [Notes en ligne] [Exemples]	532
10.54.24 pdf_set_text_matrix [Notes en ligne] [Exemples]	532
10.54.25 pdf_set_text_pos [Notes en ligne] [Exemples]	532
10.54.26 pdf_set_char_spacing [Notes en ligne] [Exemples]	532
10.54.27 pdf_set_word_spacing [Notes en ligne] [Exemples]	533
10.54.28 pdf_skew [Notes en ligne] [Exemples]	533
10.54.29 pdf_continue_text [Notes en ligne] [Exemples]	533
10.54.30 pdf_stringwidth [Notes en ligne] [Exemples]	533
10.54.31 pdf_save [Notes en ligne] [Exemples]	533
10.54.32 pdf_restore [Notes en ligne] [Exemples]	534
10.54.33 pdf_translate [Notes en ligne] [Exemples]	534
10.54.34 pdf_scale [Notes en ligne] [Exemples]	534
10.54.35 pdf_rotate [Notes en ligne] [Exemples]	535
10.54.36 pdf_setflat [Notes en ligne] [Exemples]	535
10.54.37 pdf_setlinejoin [Notes en ligne] [Exemples]	535
10.54.38 pdf_setlinecap [Notes en ligne] [Exemples]	535
10.54.39 pdf_setmiterlimit [Notes en ligne] [Exemples]	535
10.54.40 pdf_setlinewidth [Notes en ligne] [Exemples]	536
10.54.41 pdf_setdash [Notes en ligne] [Exemples]	536
10.54.42 pdf_moveto [Notes en ligne] [Exemples]	536
10.54.43 pdf_curveto [Notes en ligne] [Exemples]	536
10.54.44 pdf_lineto [Notes en ligne] [Exemples]	536
10.54.45 pdf_circle [Notes en ligne] [Exemples]	537
10.54.46 pdf_arc [Notes en ligne] [Exemples]	537
10.54.47 pdf_rect [Notes en ligne] [Exemples]	537
10.54.48 pdf_closepath [Notes en ligne] [Exemples]	537
10.54.49 pdf_stroke [Notes en ligne] [Exemples]	537
10.54.50 pdf_closepath_stroke [Notes en ligne] [Exemples]	538
10.54.51 pdf_fill [Notes en ligne] [Exemples]	538

Table of Contents

10.54.52 pdf fill stroke [Notes en ligne] [Exemples].....	538
10.54.53 pdf closepath fill stroke [Notes en ligne] [Exemples].....	538
10.54.54 pdf endpath [Notes en ligne] [Exemples].....	539
10.54.55 pdf clip [Notes en ligne] [Exemples].....	539
10.54.56 pdf setgray fill [Notes en ligne] [Exemples].....	539
10.54.57 pdf setgray stroke [Notes en ligne] [Exemples].....	539
10.54.58 pdf setgray [Notes en ligne] [Exemples].....	539
10.54.59 pdf setrgbcolor fill [Notes en ligne] [Exemples].....	540
10.54.60 pdf setrgbcolor stroke [Notes en ligne] [Exemples].....	540
10.54.61 pdf setrgbcolor [Notes en ligne] [Exemples].....	540
10.54.62 pdf add_outline [Notes en ligne] [Exemples].....	540
10.54.63 pdf set_transition [Notes en ligne] [Exemples].....	540
10.54.64 pdf set_duration [Notes en ligne] [Exemples].....	541
10.54.65 pdf open gif [Notes en ligne] [Exemples].....	541
10.54.66 pdf open png [Notes en ligne] [Exemples].....	541
10.54.67 pdf open image file [Notes en ligne] [Exemples].....	542
10.54.68 pdf open memory image [Notes en ligne] [Exemples].....	542
10.54.69 pdf open jpeg [Notes en ligne] [Exemples].....	543
10.54.70 pdf open tiff [Notes en ligne] [Exemples].....	543
10.54.71 pdf close_image [Notes en ligne] [Exemples].....	543
10.54.72 pdf place_image [Notes en ligne] [Exemples].....	543
10.54.73 pdf put_image [Notes en ligne] [Exemples].....	544
10.54.74 pdf execute_image [Notes en ligne] [Exemples].....	544
10.54.75 pdf add_annotation [Notes en ligne] [Exemples].....	545
10.54.76 pdf set_border_style [Notes en ligne] [Exemples].....	545
10.54.77 pdf set_border_color [Notes en ligne] [Exemples].....	545
10.54.78 pdf set_border_dash [Notes en ligne] [Exemples].....	545
10.55 Verisign Payflow Pro Paiement [Notes en ligne].....	545
10.55.1 pfpro_init [Notes en ligne] [Exemples].....	546
10.55.2 pfpro_cleanup [Notes en ligne] [Exemples].....	546
10.55.3 pfpro_process [Notes en ligne] [Exemples].....	546
10.55.4 pfpro_process_raw [Notes en ligne] [Exemples].....	547
10.55.5 pfpro_version [Notes en ligne] [Exemples].....	548
10.56 PostgreSQL [Notes en ligne].....	548
10.56.1 pg_close [Notes en ligne] [Exemples].....	549
10.56.2 pg_cmdtuples [Notes en ligne] [Exemples].....	549
10.56.3 pg_connect [Notes en ligne] [Exemples].....	550
10.56.4 pg_dbname [Notes en ligne] [Exemples].....	550
10.56.5 pg_end_copy [Notes en ligne] [Exemples].....	550
10.56.6 pg_errormessage [Notes en ligne] [Exemples].....	551
10.56.7 pg_exec [Notes en ligne] [Exemples].....	551
10.56.8 pg_fetch_array [Notes en ligne] [Exemples].....	551
10.56.9 pg_fetch_object [Notes en ligne] [Exemples].....	552
10.56.10 pg_fetch_row [Notes en ligne] [Exemples].....	553
10.56.11 pg_fieldisnull [Notes en ligne] [Exemples].....	554
10.56.12 pg_fieldname [Notes en ligne] [Exemples].....	554
10.56.13 pg_fieldnum [Notes en ligne] [Exemples].....	554

Table of Contents

10.56.14	pg_fieldprtlen [Notes en ligne] [Exemples]	554
10.56.15	pg_fieldsize [Notes en ligne] [Exemples]	554
10.56.16	pg_fieldtype [Notes en ligne] [Exemples]	555
10.56.17	pg_freeresult [Notes en ligne] [Exemples]	555
10.56.18	pg_getlastoid [Notes en ligne] [Exemples]	555
10.56.19	pg_host [Notes en ligne] [Exemples]	555
10.56.20	pg_loclose [Notes en ligne] [Exemples]	555
10.56.21	pg_locreate [Notes en ligne] [Exemples]	556
10.56.22	pg_loexport [Notes en ligne] [Exemples]	556
10.56.23	pg_loimport [Notes en ligne] [Exemples]	556
10.56.24	pg_loopen [Notes en ligne] [Exemples]	556
10.56.25	pg_loread [Notes en ligne] [Exemples]	556
10.56.26	pg_loreadall [Notes en ligne] [Exemples]	557
10.56.27	pg_lounlink [Notes en ligne] [Exemples]	557
10.56.28	pg_lowrite [Notes en ligne] [Exemples]	557
10.56.29	pg_numfields [Notes en ligne] [Exemples]	557
10.56.30	pg_numrows [Notes en ligne] [Exemples]	557
10.56.31	pg_options [Notes en ligne] [Exemples]	558
10.56.32	pg_pconnect [Notes en ligne] [Exemples]	558
10.56.33	pg_port [Notes en ligne] [Exemples]	558
10.56.34	pg_put_line [Notes en ligne] [Exemples]	558
10.56.35	pg_result [Notes en ligne] [Exemples]	559
10.56.36	pg_set_client_encoding [Notes en ligne] [Exemples]	559
10.56.37	pg_client_encoding [Notes en ligne] [Exemples]	559
10.56.38	pg_trace [Notes en ligne] [Exemples]	560
10.56.39	pg_tty [Notes en ligne] [Exemples]	560
10.56.40	pg_untrace [Notes en ligne] [Exemples]	560
10.57	POSIX [Notes en ligne]	560
10.57.1	posix_kill [Notes en ligne] [Exemples]	561
10.57.2	posix_getpid [Notes en ligne] [Exemples]	561
10.57.3	posix_getppid [Notes en ligne] [Exemples]	561
10.57.4	posix_getuid [Notes en ligne] [Exemples]	561
10.57.5	posix_geteuid [Notes en ligne] [Exemples]	561
10.57.6	posix_getgid [Notes en ligne] [Exemples]	562
10.57.7	posix_getegid [Notes en ligne] [Exemples]	562
10.57.8	posix_setuid [Notes en ligne] [Exemples]	562
10.57.9	posix_setgid [Notes en ligne] [Exemples]	562
10.57.10	posix_getgroups [Notes en ligne] [Exemples]	562
10.57.11	posix_getlogin [Notes en ligne] [Exemples]	563
10.57.12	posix_getpgrp [Notes en ligne] [Exemples]	563
10.57.13	posix_setsid [Notes en ligne] [Exemples]	563
10.57.14	posix_setpgid [Notes en ligne] [Exemples]	563
10.57.15	posix_getpgid [Notes en ligne] [Exemples]	563
10.57.16	posix_getsid [Notes en ligne] [Exemples]	564
10.57.17	posix_uname [Notes en ligne] [Exemples]	564
10.57.18	posix_times [Notes en ligne] [Exemples]	564
10.57.19	posix_ctermid [Notes en ligne] [Exemples]	564

Table of Contents

10.57.20	posix_ttyname [Notes en ligne] [Exemples].....	565
10.57.21	posix_isatty [Notes en ligne] [Exemples].....	565
10.57.22	posix_getcwd [Notes en ligne] [Exemples].....	565
10.57.23	posix_mkfifo [Notes en ligne] [Exemples].....	565
10.57.24	posix_getgrnam [Notes en ligne] [Exemples].....	565
10.57.25	posix_getgrgid [Notes en ligne] [Exemples].....	566
10.57.26	posix_getpwnam [Notes en ligne] [Exemples].....	566
10.57.27	posix_getpwuid [Notes en ligne] [Exemples].....	567
10.57.28	posix_getrlimit [Notes en ligne] [Exemples].....	568
10.58	Pspell [Notes en ligne].....	568
10.58.1	pspell_add_to_personal [Notes en ligne] [Exemples].....	568
10.58.2	pspell_add_to_session [Notes en ligne] [Exemples].....	568
10.58.3	pspell_check [Notes en ligne] [Exemples].....	569
10.58.4	pspell_clear_session [Notes en ligne] [Exemples].....	569
10.58.5	pspell_config_create [Notes en ligne] [Exemples].....	570
10.58.6	pspell_config_ignore [Notes en ligne] [Exemples].....	570
10.58.7	pspell_config_mode [Notes en ligne] [Exemples].....	571
10.58.8	pspell_config_personal [Notes en ligne] [Exemples].....	571
10.58.9	pspell_config_repl [Notes en ligne] [Exemples].....	572
10.58.10	pspell_config_runtogether [Notes en ligne] [Exemples].....	572
10.58.11	pspell_config_save_repl [Notes en ligne] [Exemples].....	573
10.58.12	pspell_new [Notes en ligne] [Exemples].....	573
10.58.13	pspell_new_config [Notes en ligne] [Exemples].....	574
10.58.14	pspell_new_personal [Notes en ligne] [Exemples].....	574
10.58.15	pspell_save_wordlist [Notes en ligne] [Exemples].....	575
10.58.16	pspell_store_replacement [Notes en ligne] [Exemples].....	575
10.58.17	pspell_suggest [Notes en ligne] [Exemples].....	576
10.59	Readline GNU [Notes en ligne].....	576
10.59.1	readline [Notes en ligne] [Exemples].....	577
10.59.2	readline_add_history [Notes en ligne] [Exemples].....	577
10.59.3	readline_clear_history [Notes en ligne] [Exemples].....	577
10.59.4	readline_completion_function [Notes en ligne] [Exemples].....	577
10.59.5	readline_info [Notes en ligne] [Exemples].....	578
10.59.6	readline_list_history [Notes en ligne] [Exemples].....	578
10.59.7	readline_read_history [Notes en ligne] [Exemples].....	578
10.59.8	readline_write_history [Notes en ligne] [Exemples].....	578
10.60	Recode (GNU) [Notes en ligne].....	578
10.60.1	recode_string [Notes en ligne] [Exemples].....	579
10.60.2	recode [Notes en ligne] [Exemples].....	579
10.60.3	recode_file [Notes en ligne] [Exemples].....	579
10.61	Expressions régulières [Notes en ligne].....	579
10.61.1	ereg [Notes en ligne] [Exemples].....	580
10.61.2	ereg_replace [Notes en ligne] [Exemples].....	581
10.61.3	eregi [Notes en ligne] [Exemples].....	582
10.61.4	eregi_replace [Notes en ligne] [Exemples].....	582
10.61.5	split [Notes en ligne] [Exemples].....	582
10.61.6	spliti [Notes en ligne] [Exemples].....	583

Table of Contents

10.61.7	sql_regcase [Notes en ligne] [Exemples]	583
10.62	Satellite CORBA client extension [Notes en ligne]	584
10.62.1	orbitobject [Notes en ligne]	584
10.62.2	orbitenum [Notes en ligne]	584
10.62.3	orbitstruct [Notes en ligne]	585
10.62.4	satellite_caught_exception [Notes en ligne] [Exemples]	585
10.62.5	satellite_exception_id [Notes en ligne] [Exemples]	586
10.62.6	satellite_exception_value [Notes en ligne] [Exemples]	586
10.63	Sémaphores et gestion de la mémoire partagée [Notes en ligne]	587
10.63.1	sem_get [Notes en ligne] [Exemples]	587
10.63.2	sem_acquire [Notes en ligne] [Exemples]	587
10.63.3	sem_release [Notes en ligne] [Exemples]	588
10.63.4	shm_attach [Notes en ligne] [Exemples]	588
10.63.5	shm_detach [Notes en ligne] [Exemples]	588
10.63.6	shm_remove [Notes en ligne] [Exemples]	588
10.63.7	shm_put_var [Notes en ligne] [Exemples]	589
10.63.8	shm_get_var [Notes en ligne] [Exemples]	589
10.63.9	shm_remove_var [Notes en ligne] [Exemples]	589
10.64	SESAM [Notes en ligne]	589
10.64.1	sesam_connect [Notes en ligne] [Exemples]	593
10.64.2	sesam_disconnect [Notes en ligne] [Exemples]	594
10.64.3	sesam_settransaction [Notes en ligne] [Exemples]	594
10.64.4	sesam_commit [Notes en ligne] [Exemples]	595
10.64.5	sesam_rollback [Notes en ligne] [Exemples]	595
10.64.6	sesam_execimm [Notes en ligne] [Exemples]	596
10.64.7	sesam_query [Notes en ligne] [Exemples]	596
10.64.8	sesam_num_fields [Notes en ligne] [Exemples]	598
10.64.9	sesam_field_name [Notes en ligne] [Exemples]	598
10.64.10	sesam_diagnostic [Notes en ligne] [Exemples]	598
10.64.11	sesam_fetch_result [Notes en ligne] [Exemples]	600
10.64.12	sesam_affected_rows [Notes en ligne] [Exemples]	601
10.64.13	sesam_errormsg [Notes en ligne] [Exemples]	601
10.64.14	sesam_field_array [Notes en ligne] [Exemples]	601
10.64.15	sesam_fetch_row [Notes en ligne] [Exemples]	603
10.64.16	sesam_fetch_array [Notes en ligne] [Exemples]	605
10.64.17	sesam_seek_row [Notes en ligne] [Exemples]	606
10.64.18	sesam_free_result [Notes en ligne] [Exemples]	606
10.65	Sessions [Notes en ligne]	607
10.65.1	session_start [Notes en ligne] [Exemples]	609
10.65.2	session_destroy [Notes en ligne] [Exemples]	609
10.65.3	session_name [Notes en ligne] [Exemples]	609
10.65.4	session_module_name [Notes en ligne] [Exemples]	610
10.65.5	session_save_path [Notes en ligne] [Exemples]	610
10.65.6	session_id [Notes en ligne] [Exemples]	610
10.65.7	session_register [Notes en ligne] [Exemples]	611
10.65.8	session_unregister [Notes en ligne] [Exemples]	611
10.65.9	session_unset [Notes en ligne] [Exemples]	611

Table of Contents

10.65.10 session_is_registered [Notes en ligne] [Exemples].....	611
10.65.11 session_get_cookie_params [Notes en ligne] [Exemples].....	611
10.65.12 session_set_cookie_params [Notes en ligne] [Exemples].....	612
10.65.13 session_decode [Notes en ligne] [Exemples].....	612
10.65.14 session_encode [Notes en ligne] [Exemples].....	612
10.65.15 session_set_save_handler [Notes en ligne] [Exemples].....	612
10.65.16 session_cache_limiter [Notes en ligne] [Exemples].....	614
10.66 Mémoire partagée [Notes en ligne]	614
10.66.1 shm_open [Notes en ligne] [Exemples].....	615
10.66.2 shm_read [Notes en ligne] [Exemples].....	615
10.66.3 shm_write [Notes en ligne] [Exemples].....	616
10.66.4 shm_size [Notes en ligne] [Exemples].....	616
10.66.5 shm_delete [Notes en ligne] [Exemples].....	617
10.66.6 shm_close [Notes en ligne] [Exemples].....	617
10.67 SNMP [Notes en ligne]	617
10.67.1 snmpget [Notes en ligne] [Exemples].....	618
10.67.2 snmpset [Notes en ligne] [Exemples].....	618
10.67.3 snmpwalk [Notes en ligne] [Exemples].....	618
10.67.4 snmpwalkoid [Notes en ligne] [Exemples].....	619
10.67.5 snmp_get_quick_print [Notes en ligne] [Exemples].....	619
10.67.6 snmp_set_quick_print [Notes en ligne] [Exemples].....	620
10.68 Sockets [Notes en ligne]	620
10.68.1 accept connect [Notes en ligne] [Exemples].....	622
10.68.2 bind [Notes en ligne] [Exemples].....	623
10.68.3 close [Notes en ligne] [Exemples].....	623
10.68.4 connect [Notes en ligne] [Exemples].....	623
10.68.5 listen [Notes en ligne] [Exemples].....	624
10.68.6 read [Notes en ligne] [Exemples].....	624
10.68.7 socket [Notes en ligne] [Exemples].....	624
10.68.8 strerror [Notes en ligne] [Exemples].....	625
10.68.9 write [Notes en ligne] [Exemples].....	625
10.69 Chaîne de caractères [Notes en ligne]	625
10.69.1 addslashes [Notes en ligne] [Exemples].....	626
10.69.2 addslashes [Notes en ligne] [Exemples].....	626
10.69.3 bin2hex [Notes en ligne] [Exemples].....	626
10.69.4 chop [Notes en ligne] [Exemples].....	626
10.69.5 chr [Notes en ligne] [Exemples].....	627
10.69.6 chunk_split [Notes en ligne] [Exemples].....	627
10.69.7 convert_cyr_string [Notes en ligne] [Exemples].....	627
10.69.8 count_chars [Notes en ligne] [Exemples].....	628
10.69.9 crc32 [Notes en ligne] [Exemples].....	628
10.69.10 crypt [Notes en ligne] [Exemples].....	628
10.69.11 echo [Notes en ligne] [Exemples].....	629
10.69.12 explode [Notes en ligne] [Exemples].....	629
10.69.13 get_html_translation_table [Notes en ligne] [Exemples].....	630
10.69.14 get_meta_tags [Notes en ligne] [Exemples].....	630
10.69.15 hebrew [Notes en ligne] [Exemples].....	631

Table of Contents

10.69.16 hebrevc [Notes en ligne] [Exemples]	631
10.69.17 htmlentities [Notes en ligne] [Exemples]	631
10.69.18 htmlspecialchars [Notes en ligne] [Exemples]	632
10.69.19 implode [Notes en ligne] [Exemples]	632
10.69.20 join [Notes en ligne] [Exemples]	633
10.69.21 levenshtein [Notes en ligne] [Exemples]	633
10.69.22 localeconv [Notes en ligne] [Exemples]	634
10.69.23 ltrim [Notes en ligne] [Exemples]	636
10.69.24 md5 [Notes en ligne] [Exemples]	636
10.69.25 metaphone [Notes en ligne] [Exemples]	636
10.69.26 nl2br [Notes en ligne] [Exemples]	637
10.69.27 ord [Notes en ligne] [Exemples]	637
10.69.28 parse_str [Notes en ligne] [Exemples]	637
10.69.29 print [Notes en ligne] [Exemples]	638
10.69.30 printf [Notes en ligne] [Exemples]	638
10.69.31 quoted_printable_decode [Notes en ligne] [Exemples]	638
10.69.32 quotemeta [Notes en ligne] [Exemples]	638
10.69.33 rtrim [Notes en ligne] [Exemples]	638
10.69.34 sscanf [Notes en ligne] [Exemples]	639
10.69.35 setlocale [Notes en ligne] [Exemples]	639
10.69.36 similar_text [Notes en ligne] [Exemples]	640
10.69.37 soundex [Notes en ligne] [Exemples]	640
10.69.38 sprintf [Notes en ligne] [Exemples]	641
10.69.39 strncasecmp [Notes en ligne] [Exemples]	642
10.69.40 strcasecmp [Notes en ligne] [Exemples]	642
10.69.41 strchr [Notes en ligne] [Exemples]	642
10.69.42 strcmp [Notes en ligne] [Exemples]	643
10.69.43 strcoll [Notes en ligne] [Exemples]	643
10.69.44 strcspn [Notes en ligne] [Exemples]	643
10.69.45 strip_tags [Notes en ligne] [Exemples]	643
10.69.46 stripslashes [Notes en ligne] [Exemples]	644
10.69.47 stripslashes [Notes en ligne] [Exemples]	644
10.69.48 stristr [Notes en ligne] [Exemples]	644
10.69.49 strlen [Notes en ligne] [Exemples]	644
10.69.50 strnatcmp [Notes en ligne] [Exemples]	644
10.69.51 strnatcasecmp [Notes en ligne] [Exemples]	645
10.69.52 strncmp [Notes en ligne] [Exemples]	645
10.69.53 str_pad [Notes en ligne] [Exemples]	646
10.69.54 strpos [Notes en ligne] [Exemples]	646
10.69.55 strrchr [Notes en ligne] [Exemples]	647
10.69.56 str_repeat [Notes en ligne] [Exemples]	647
10.69.57 strev [Notes en ligne] [Exemples]	647
10.69.58 strrpos [Notes en ligne] [Exemples]	648
10.69.59 strspn [Notes en ligne] [Exemples]	648
10.69.60 strstr [Notes en ligne] [Exemples]	649
10.69.61 strtok [Notes en ligne] [Exemples]	649
10.69.62 strtolower [Notes en ligne] [Exemples]	650

Table of Contents

10.69.63	strtoupper [Notes en ligne] [Exemples]	650
10.69.64	str_replace [Notes en ligne] [Exemples]	650
10.69.65	strtr [Notes en ligne] [Exemples]	651
10.69.66	substr [Notes en ligne] [Exemples]	652
10.69.67	substr_count [Notes en ligne] [Exemples]	652
10.69.68	substr_replace [Notes en ligne] [Exemples]	653
10.69.69	trim [Notes en ligne] [Exemples]	653
10.69.70	ucfirst [Notes en ligne] [Exemples]	654
10.69.71	ucwords [Notes en ligne] [Exemples]	654
10.69.72	wordwrap [Notes en ligne] [Exemples]	654
10.70	Shockwave Flash [Notes en ligne]	655
10.70.1	swf_openfile [Notes en ligne] [Exemples]	656
10.70.2	swf_closefile [Notes en ligne] [Exemples]	656
10.70.3	swf_labelframe [Notes en ligne] [Exemples]	657
10.70.4	swf_showframe [Notes en ligne] [Exemples]	658
10.70.5	swf_setframe [Notes en ligne] [Exemples]	658
10.70.6	swf_getframe [Notes en ligne] [Exemples]	658
10.70.7	swf_mulcolor [Notes en ligne] [Exemples]	658
10.70.8	swf_addcolor [Notes en ligne] [Exemples]	658
10.70.9	swf_placeobject [Notes en ligne] [Exemples]	659
10.70.10	swf_modifyobject [Notes en ligne] [Exemples]	659
10.70.11	swf_removeobject [Notes en ligne] [Exemples]	659
10.70.12	swf_nextid [Notes en ligne] [Exemples]	659
10.70.13	swf_startdoaction [Notes en ligne] [Exemples]	659
10.70.14	swf_actiongotoframe [Notes en ligne] [Exemples]	660
10.70.15	swf_actiongeturl [Notes en ligne] [Exemples]	660
10.70.16	swf_actionnextframe [Notes en ligne] [Exemples]	660
10.70.17	swf_actionprevframe [Notes en ligne] [Exemples]	660
10.70.18	swf_actionplay [Notes en ligne] [Exemples]	660
10.70.19	swf_actionstop [Notes en ligne] [Exemples]	660
10.70.20	swf_actiontogglequality [Notes en ligne] [Exemples]	661
10.70.21	swf_actionwaitforframe [Notes en ligne] [Exemples]	661
10.70.22	swf_actionsettarget [Notes en ligne] [Exemples]	661
10.70.23	swf_actiongotolabel [Notes en ligne] [Exemples]	661
10.70.24	swf_enddoaction [Notes en ligne] [Exemples]	661
10.70.25	swf_defineline [Notes en ligne] [Exemples]	662
10.70.26	swf_definerect [Notes en ligne] [Exemples]	662
10.70.27	swf_definepoly [Notes en ligne] [Exemples]	662
10.70.28	swf_startshape [Notes en ligne] [Exemples]	662
10.70.29	swf_shapelinesolid [Notes en ligne] [Exemples]	662
10.70.30	swf_shapefilloff [Notes en ligne] [Exemples]	663
10.70.31	swf_shapefillsolid [Notes en ligne] [Exemples]	663
10.70.32	swf_shapefillbitmapclip [Notes en ligne] [Exemples]	663
10.70.33	swf_shapefillbitmaptile [Notes en ligne] [Exemples]	663
10.70.34	swf_shapemoveto [Notes en ligne] [Exemples]	663
10.70.35	swf_shapelineto [Notes en ligne] [Exemples]	663
10.70.36	swf_shapecurveto [Notes en ligne] [Exemples]	664

Table of Contents

10.70.37 swf_shapecurveto3 [Notes en ligne] [Exemples]	664
10.70.38 swf_shapearc [Notes en ligne] [Exemples]	664
10.70.39 swf_endshape [Notes en ligne] [Exemples]	664
10.70.40 swf_definefont [Notes en ligne] [Exemples]	664
10.70.41 swf_setfont [Notes en ligne] [Exemples]	665
10.70.42 swf_fontsize [Notes en ligne] [Exemples]	665
10.70.43 swf_fontslant [Notes en ligne] [Exemples]	665
10.70.44 swf_fontracking [Notes en ligne] [Exemples]	665
10.70.45 swf_getfontinfo [Notes en ligne] [Exemples]	665
10.70.46 swf_definetext [Notes en ligne] [Exemples]	666
10.70.47 swf_textwidth [Notes en ligne] [Exemples]	666
10.70.48 swf_definebitmap [Notes en ligne] [Exemples]	666
10.70.49 swf_getbitmapinfo [Notes en ligne] [Exemples]	666
10.70.50 swf_startsymbol [Notes en ligne] [Exemples]	666
10.70.51 swf_endsymbol [Notes en ligne] [Exemples]	667
10.70.52 swf_startbutton [Notes en ligne] [Exemples]	667
10.70.53 swf_addbuttonrecord [Notes en ligne] [Exemples]	667
10.70.54 swf_oncondition [Notes en ligne] [Exemples]	667
10.70.55 swf_endbutton [Notes en ligne] [Exemples]	668
10.70.56 swf_viewport [Notes en ligne] [Exemples]	668
10.70.57 swf_ortho [Notes en ligne] [Exemples]	668
10.70.58 swf_ortho2 [Notes en ligne] [Exemples]	669
10.70.59 swf_perspective [Notes en ligne] [Exemples]	669
10.70.60 swf_polarview [Notes en ligne] [Exemples]	669
10.70.61 swf_lookat [Notes en ligne] [Exemples]	669
10.70.62 swf_pushmatrix [Notes en ligne] [Exemples]	670
10.70.63 swf_popmatrix [Notes en ligne] [Exemples]	670
10.70.64 swf_scale [Notes en ligne] [Exemples]	670
10.70.65 swf_translate [Notes en ligne] [Exemples]	670
10.70.66 swf_rotate [Notes en ligne] [Exemples]	670
10.70.67 swf_posround [Notes en ligne] [Exemples]	671
10.71 Sybase [Notes en ligne]	671
10.71.1 sybase_affected_rows [Notes en ligne] [Exemples]	671
10.71.2 sybase_close [Notes en ligne] [Exemples]	671
10.71.3 sybase_connect [Notes en ligne] [Exemples]	671
10.71.4 sybase_data_seek [Notes en ligne] [Exemples]	672
10.71.5 sybase_fetch_array [Notes en ligne] [Exemples]	672
10.71.6 sybase_fetch_field [Notes en ligne] [Exemples]	672
10.71.7 sybase_fetch_object [Notes en ligne] [Exemples]	673
10.71.8 sybase_fetch_row [Notes en ligne] [Exemples]	673
10.71.9 sybase_field_seek [Notes en ligne] [Exemples]	673
10.71.10 sybase_free_result [Notes en ligne] [Exemples]	674
10.71.11 sybase_get_last_message [Notes en ligne] [Exemples]	674
10.71.12 sybase_min_client_severity [Notes en ligne] [Exemples]	674
10.71.13 sybase_min_error_severity [Notes en ligne] [Exemples]	674
10.71.14 sybase_min_message_severity [Notes en ligne] [Exemples]	674
10.71.15 sybase_min_server_severity [Notes en ligne] [Exemples]	675

Table of Contents

10.71.16 sybase_num_fields [Notes en ligne] [Exemples]	675
10.71.17 sybase_num_rows [Notes en ligne] [Exemples]	675
10.71.18 sybase_pconnect [Notes en ligne] [Exemples]	675
10.71.19 sybase_query [Notes en ligne] [Exemples]	676
10.71.20 sybase_result [Notes en ligne] [Exemples]	676
10.71.21 sybase_select_db [Notes en ligne] [Exemples]	676
10.72 ODBC unifié [Notes en ligne]	676
10.72.1 odbc_autocommit [Notes en ligne] [Exemples]	677
10.72.2 odbc_binmode [Notes en ligne] [Exemples]	677
10.72.3 odbc_close [Notes en ligne] [Exemples]	678
10.72.4 odbc_close_all [Notes en ligne] [Exemples]	678
10.72.5 odbc_commit [Notes en ligne] [Exemples]	678
10.72.6 odbc_connect [Notes en ligne] [Exemples]	678
10.72.7 odbc_cursor [Notes en ligne] [Exemples]	679
10.72.8 odbc_do [Notes en ligne] [Exemples]	679
10.72.9 odbc_error [Notes en ligne] [Exemples]	679
10.72.10 odbc_errormsg [Notes en ligne] [Exemples]	679
10.72.11 odbc_exec [Notes en ligne] [Exemples]	680
10.72.12 odbc_execute [Notes en ligne] [Exemples]	680
10.72.13 odbc_fetch_into [Notes en ligne] [Exemples]	680
10.72.14 odbc_fetch_row [Notes en ligne] [Exemples]	680
10.72.15 odbc_field_name [Notes en ligne] [Exemples]	681
10.72.16 odbc_field_num [Notes en ligne] [Exemples]	681
10.72.17 odbc_field_type [Notes en ligne] [Exemples]	681
10.72.18 odbc_field_len [Notes en ligne] [Exemples]	681
10.72.19 odbc_field_precision [Notes en ligne] [Exemples]	681
10.72.20 odbc_field_scale [Notes en ligne] [Exemples]	682
10.72.21 odbc_free_result [Notes en ligne] [Exemples]	682
10.72.22 odbc_longreadlen [Notes en ligne] [Exemples]	682
10.72.23 odbc_num_fields [Notes en ligne] [Exemples]	682
10.72.24 odbc_pconnect [Notes en ligne] [Exemples]	683
10.72.25 odbc_prepare [Notes en ligne] [Exemples]	683
10.72.26 odbc_num_rows [Notes en ligne] [Exemples]	683
10.72.27 odbc_result [Notes en ligne] [Exemples]	683
10.72.28 odbc_result_all [Notes en ligne] [Exemples]	684
10.72.29 odbc_rollback [Notes en ligne] [Exemples]	684
10.72.30 odbc_setoption [Notes en ligne] [Exemples]	684
10.72.31 odbc_tables [Notes en ligne] [Exemples]	685
10.72.32 odbc_tableprivileges [Notes en ligne] [Exemples]	686
10.72.33 odbc_columns [Notes en ligne] [Exemples]	686
10.72.34 odbc_columnprivileges [Notes en ligne] [Exemples]	687
10.72.35 odbc_gettypeinfo [Notes en ligne] [Exemples]	687
10.72.36 odbc_primarykeys [Notes en ligne] [Exemples]	688
10.72.37 odbc_foreignkeys [Notes en ligne] [Exemples]	688
10.72.38 odbc_procedures [Notes en ligne] [Exemples]	689
10.72.39 odbc_procedurecolumns [Notes en ligne] [Exemples]	690
10.72.40 odbc_specialcolumns [Notes en ligne] [Exemples]	690

Table of Contents

10.72.41	odbc_statistics [Notes en ligne] [Exemples]	691
10.73	URL [Notes en ligne]	691
10.73.1	base64_decode [Notes en ligne] [Exemples]	691
10.73.2	base64_encode [Notes en ligne] [Exemples]	692
10.73.3	parse_url [Notes en ligne] [Exemples]	692
10.73.4	rawurldecode [Notes en ligne] [Exemples]	692
10.73.5	rawurlencode [Notes en ligne] [Exemples]	692
10.73.6	urldecode [Notes en ligne] [Exemples]	693
10.73.7	urlencode [Notes en ligne] [Exemples]	693
10.74	Variables [Notes en ligne]	694
10.74.1	doubleval [Notes en ligne] [Exemples]	694
10.74.2	empty [Notes en ligne] [Exemples]	694
10.74.3	gettype [Notes en ligne] [Exemples]	695
10.74.4	intval [Notes en ligne] [Exemples]	695
10.74.5	is_array [Notes en ligne] [Exemples]	696
10.74.6	is_bool [Notes en ligne] [Exemples]	696
10.74.7	is_double [Notes en ligne] [Exemples]	696
10.74.8	is_float [Notes en ligne] [Exemples]	696
10.74.9	is_int [Notes en ligne] [Exemples]	696
10.74.10	is_integer [Notes en ligne] [Exemples]	697
10.74.11	is_long [Notes en ligne] [Exemples]	697
10.74.12	is_numeric [Notes en ligne] [Exemples]	697
10.74.13	is_object [Notes en ligne] [Exemples]	697
10.74.14	is_real [Notes en ligne] [Exemples]	697
10.74.15	is_resource [Notes en ligne] [Exemples]	698
10.74.16	is_string [Notes en ligne] [Exemples]	698
10.74.17	isset [Notes en ligne] [Exemples]	698
10.74.18	print_r [Notes en ligne] [Exemples]	698
10.74.19	serialize [Notes en ligne] [Exemples]	699
10.74.20	settype [Notes en ligne] [Exemples]	699
10.74.21	strval [Notes en ligne] [Exemples]	700
10.74.22	unserialize [Notes en ligne] [Exemples]	700
10.74.23	unset [Notes en ligne] [Exemples]	701
10.74.24	var_dump [Notes en ligne] [Exemples]	702
10.75	WDDX [Notes en ligne]	702
10.75.1	wddx_serialize_value [Notes en ligne] [Exemples]	703
10.75.2	wddx_serialize_vars [Notes en ligne] [Exemples]	703
10.75.3	wddx_packet_start [Notes en ligne] [Exemples]	704
10.75.4	wddx_packet_end [Notes en ligne] [Exemples]	704
10.75.5	wddx_add_vars [Notes en ligne] [Exemples]	704
10.75.6	wddx_deserialize [Notes en ligne] [Exemples]	705
10.76	Analyseur syntaxique XML [Notes en ligne]	705
10.76.1	Introduction [Notes en ligne]	705
10.76.1.1	A propos de XML [Notes en ligne]	705
10.76.1.2	Installation [Notes en ligne]	705
10.76.1.3	A propos de cette extension : [Notes en ligne]	705
10.76.1.4	Problèmes de casse [Notes en ligne]	706

Table of Contents

10.76.1.5	Error Codes [Notes en ligne]	707
10.76.1.6	Codage des caractères [Notes en ligne]	707
10.76.2	Quelques exemples [Notes en ligne]	708
10.76.2.1	Exemple de structure XML [Notes en ligne]	708
10.76.2.2	XML Transtypage XML -> HTML [Notes en ligne]	708
10.76.2.3	XML Entité externe [Notes en ligne]	709
10.76.3	xml_parser_create [Notes en ligne] [Exemples]	712
10.76.4	xml_set_object [Notes en ligne] [Exemples]	712
10.76.5	xml_set_element_handler [Notes en ligne] [Exemples]	713
10.76.6	xml_set_character_data_handler [Notes en ligne] [Exemples]	714
10.76.7	xml_set_processing_instruction_handler [Notes en ligne] [Exemples]	714
10.76.8	xml_set_default_handler [Notes en ligne] [Exemples]	715
10.76.9	xml_set_unparsed_entity_decl_handler [Notes en ligne] [Exemples]	716
10.76.10	xml_set_notation_decl_handler [Notes en ligne] [Exemples]	716
10.76.11	xml_set_external_entity_ref_handler [Notes en ligne] [Exemples]	717
10.76.12	xml_parse [Notes en ligne] [Exemples]	718
10.76.13	xml_get_error_code [Notes en ligne] [Exemples]	718
10.76.14	xml_error_string [Notes en ligne] [Exemples]	719
10.76.15	xml_get_current_line_number [Notes en ligne] [Exemples]	719
10.76.16	xml_get_current_column_number [Notes en ligne] [Exemples]	719
10.76.17	xml_get_current_byte_index [Notes en ligne] [Exemples]	720
10.76.18	xml_parse_into_struct [Notes en ligne] [Exemples]	720
10.76.19	xml_parser_free [Notes en ligne] [Exemples]	722
10.76.20	xml_parser_set_option [Notes en ligne] [Exemples]	722
10.76.21	xml_parser_get_option [Notes en ligne] [Exemples]	723
10.76.22	utf8_decode [Notes en ligne] [Exemples]	723
10.76.23	utf8_encode [Notes en ligne] [Exemples]	723
10.77	XSLT [Notes en ligne]	724
10.77.1	Introduction [Notes en ligne]	724
10.77.1.1	A propos de XSLT et Sablotron [Notes en ligne]	724
10.77.1.2	Installation [Notes en ligne]	724
10.77.1.3	A propos de Sablotron [Notes en ligne]	724
10.77.2	xslt_closelog [Notes en ligne] [Exemples]	724
10.77.3	xslt_create [Notes en ligne] [Exemples]	725
10.77.4	xslt_errno [Notes en ligne] [Exemples]	725
10.77.5	xslt_error [Notes en ligne] [Exemples]	725
10.77.6	xslt_fetch_result [Notes en ligne] [Exemples]	725
10.77.7	xslt_free [Notes en ligne] [Exemples]	726
10.77.8	xslt_openlog [Notes en ligne] [Exemples]	726
10.77.9	xslt_output_begintransform [Notes en ligne] [Exemples]	726
10.77.10	xslt_output_endtransform [Notes en ligne] [Exemples]	726
10.77.11	xslt_process [Notes en ligne] [Exemples]	727
10.77.12	xslt_run [Notes en ligne] [Exemples]	728
10.77.13	xslt_set_sax_handler [Notes en ligne] [Exemples]	728
10.77.14	xslt_transform [Notes en ligne] [Exemples]	728
10.78	YAZ [Notes en ligne]	728
10.78.1	Introduction [Notes en ligne]	728

Table of Contents

10.78.2	Installation [Notes en ligne]	729
10.78.3	Exemple [Notes en ligne]	729
10.78.4	yaz_addinfo [Notes en ligne] [Exemples]	730
10.78.5	yaz_close [Notes en ligne] [Exemples]	730
10.78.6	yaz_connect [Notes en ligne] [Exemples]	730
10.78.7	yaz_errno [Notes en ligne] [Exemples]	731
10.78.8	yaz_error [Notes en ligne] [Exemples]	731
10.78.9	yaz_hits [Notes en ligne] [Exemples]	731
10.78.10	yaz_range [Notes en ligne] [Exemples]	731
10.78.11	yaz_record [Notes en ligne] [Exemples]	732
10.78.12	yaz_search [Notes en ligne] [Exemples]	732
10.78.13	yaz_syntax [Notes en ligne] [Exemples]	733
10.78.14	yaz_wait [Notes en ligne] [Exemples]	733
10.79	Zlib (Compression) [Notes en ligne]	733
10.79.1	Petit exemple [Notes en ligne]	733
10.79.2	gzclose [Notes en ligne] [Exemples]	734
10.79.3	gzeof [Notes en ligne] [Exemples]	734
10.79.4	gzfile [Notes en ligne] [Exemples]	734
10.79.5	gzgetc [Notes en ligne] [Exemples]	735
10.79.6	gzgets [Notes en ligne] [Exemples]	735
10.79.7	gzgetss [Notes en ligne] [Exemples]	735
10.79.8	gzopen [Notes en ligne] [Exemples]	735
10.79.9	gzpassthru [Notes en ligne] [Exemples]	736
10.79.10	gzputs [Notes en ligne] [Exemples]	736
10.79.11	gzread [Notes en ligne] [Exemples]	736
10.79.12	gzrewind [Notes en ligne] [Exemples]	737
10.79.13	gzseek [Notes en ligne] [Exemples]	737
10.79.14	gztell [Notes en ligne] [Exemples]	737
10.79.15	gzwrite [Notes en ligne] [Exemples]	738
10.79.16	readgzfile [Notes en ligne] [Exemples]	738
10.79.17	gzcompress [Notes en ligne] [Exemples]	738
10.79.18	gzuncompress [Notes en ligne] [Exemples]	738
A	Débugueur PHP [Notes en ligne]	740
A.1	A propos du debugueur [Notes en ligne]	740
A.2	Utiliser le débugeur PHP [Notes en ligne]	740
A.3	Protocole du débugeur [Notes en ligne]	740
B	Migration de PHP/FI 2.0 à PHP 3.0 [Notes en ligne]	743
B.1	A propos des incompatibilités en 3.0 [Notes en ligne]	743
B.2	Balises PHP [Notes en ligne]	743
B.3	if..endif syntax [Notes en ligne]	744
B.4	while syntax [Notes en ligne]	744
B.5	Types d'expression [Notes en ligne]	745
B.6	Les messages d'erreur ont changé [Notes en ligne]	745
B.7	Evaluation rapide des booléens [Notes en ligne]	746
B.8	La valeur TRUE/FALSE comme retour de fonctions [Notes en ligne]	746

Table of Contents

B.9 Diverses incompatibilités [Notes en ligne]	746
C Développement PHP [Notes en ligne]	748
C.1 Créer une fonction PHP 3 [Notes en ligne]	748
C.1.1 Prototypes de fonctions [Notes en ligne]	748
C.1.2 Arguments de fonctions [Notes en ligne]	748
C.1.3 Fonctions à nombre d'arguments variable [Notes en ligne]	748
C.1.4 Utiliser les arguments d'une fonction [Notes en ligne]	749
C.1.5 Gestion de la mémoire dans une fonction [Notes en ligne]	749
C.1.6 Affecter une variable dans la table des symboles [Notes en ligne]	750
C.1.7 Retourne une valeur simple [Notes en ligne]	752
C.1.8 Retourner des valeurs complexes [Notes en ligne]	752
C.1.9 Using the resource list [Notes en ligne]	753
C.1.10 Utiliser la table des ressources persistantes. [Notes en ligne]	754
C.1.11 Ajouter des directives de configuration à l'exécution [Notes en ligne]	755
C.2 Appeler des fonctions utilisateurs [Notes en ligne]	756
C.2.1 HashTable *function table [Notes en ligne]	756
C.2.2 pval *object [Notes en ligne]	756
C.2.3 pval *function name [Notes en ligne]	757
C.2.4 pval *retval [Notes en ligne]	757
C.2.5 int param count [Notes en ligne]	757
C.2.6 pval *params[] [Notes en ligne]	757
C.3 Rapport d'erreurs [Notes en ligne]	757
C.3.1 E NOTICE [Notes en ligne]	757
C.3.2 E WARNING [Notes en ligne]	757
C.3.3 E ERROR [Notes en ligne]	758
C.3.4 E PARSE [Notes en ligne]	758
C.3.5 E CORE ERROR [Notes en ligne]	758
C.3.6 E CORE WARNING [Notes en ligne]	758
C.3.7 E COMPILE ERROR [Notes en ligne]	758
C.3.8 E COMPILE WARNING [Notes en ligne]	758
C.3.9 E USER ERROR [Notes en ligne]	758
C.3.10 E USER WARNING [Notes en ligne]	758
C.3.11 E USER NOTICE [Notes en ligne]	759
Index des fonctions [Notes en ligne] [Exemples]	760
A	760
B	761
C	761
D	763
E	764
F	765
G	767
H	769
I	770
J	774
K	775

Table of Contents

L	775
M	776
N	779
O	780
P	782
Q	786
R	786
S	787
T	791
U	791
V	792
W	792
X	792
Y	793
Z	793
Index des concepts [Notes en ligne]	794
A	794
B	796
C	796
D	800
E	802
F	805
G	807
H	808
I	808
J	809
L	809
M	812
N	813
O	813
P	814
R	815
S	823
T	824
U	825
V	825
Index des exemples [Notes en ligne]	826
A	826
B	826
C	826
D	827
E	827
F	832
G	832
I	832

Table of Contents

L	833
M	833
N	834
O	834
P	834
Q	834
R	835
S	835
T	835
U	835
V	836
W	836
X	836
Y	836

1 Preface

[\[Notes en ligne\]](#)

1.1 Les auteurs

[\[Notes en ligne\]](#)

Stig Sæther Bakken
Alexander Aulbach
Egon Schmid
Jim Winstead
Lars Torben Wilson
Rasmus Lerdorf
Zeev Suraski
Andrei Zmievski
Jouni Ahto

Editeur : Damien Seguy 1997 1998 1999 2000 2001 *PHP Documentation Group*

1.2 Copyright

[\[Notes en ligne\]](#)

Ce manuel est © Copyright 1997, 1998, 1999, 2000, 2001 par PHP Documentation Group. Les membres de ce groupe sont listés [1.1 Les auteurs](#).

Ce manuel peut être redistribué sous licence GNU General Public License, comme stipulé par la Free Software Foundation; soit la version 2 de la Licence, soit (à votre choix), une version ultérieure.

2 Préface

[\[Notes en ligne\]](#)

PHP, signifie "PHP: Hypertext Preprocessor" (Préprocesseur HyperTexte) , est un langage de script HTML. L'essentiel de sa syntaxe est empruntée aux langages C, Java et Perl, mais y ajoute plusieurs fonctionnalités uniques. Le but de ce langage est de permettre aux développeurs web de concevoir rapidement des sites aux pages dynamiques

2.1 A propos de ce manuel

[\[Notes en ligne\]](#)

Ce manuel est écrit en **SGML** en utilisant [DocBook DTD](#), avec [DSSSL](#) (Document Style and Semantics Specification Language) pour le formatage. Les utilitaires utilisés pour générer les formats **HTML**, **TeX** et **RTF** sont [Jade](#), écrit par [James Clark](#) et [The Modular DocBook Stylesheets](#) écrit par [Norman Walsh](#). La documentation PHP a été assemblée par <http://cvs.php.net/>.

Vous pouvez télécharger le manuel en diverses langues et formats, y compris **PDF**, texte simple, **HTML**, WinHelp, et **RTF** à l'adresse : <http://www.php.net/docs.php>.

Le manuel est mis à jour tous les jours, y compris les traductions. Il est disponible à l'adresse <http://snaps.php.net/manual/>. Ce manuel a été traduit en Français par de bibi@php.net. La version française est disponible chez Nexen nexen.net/. Il a été généré à partir de la documentation originale en anglais du PHP documentation Group, au format XML, grâce à une version adaptée de [texi](#).

Vous pouvez trouver plus d'informations sur comment télécharger le code de cette documentation **XML** à <http://cvs.php.net/>. La documentation est stockées dans le module phpdoc.

3 Copyright, distribution, historique

[\[Notes en ligne\]](#)

PHP est copyright (C) 1997 par PHP Development Team. Les membres de cette équipe sont listés dans le fichier de crédits, fourni avec la distribution source de PHP 3.0.

PHP est un logiciel libre : vous pouvez le modifier et/ou le modifier sous licence GNU General Public License, telle que publiée par Free Software Foundation; utilisez soit la version 2 de la License, ou toute version ultérieure (à votre convenance).

PHP est distribué avec l'espoir qu'il sera utile, mais SANS AUCUNE GARANTIE; sans même la garantie de COMMERCIALISATION ou d'UTILITE POUR UN BUT QUELCONQUE. Reportez vous à la licence GNU General Public License pour plus de détails.

Vous devez avoir reçu une copie de la GNU General Public License avec ce programme; sinon, écrivez à la fondation Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

4 Installation

[\[Notes en ligne\]](#)

4.1 Télécharger la dernière version

[\[Notes en ligne\]](#)

Le code source ainsi que des binaires pour certaines plates-formes (notamment Windows), sont disponibles à l'adresse suivante: <http://www.php.net/>.

4.2 Installation sous UNIX

[\[Notes en ligne\]](#)

Ce chapitre va vous aider lors de la configuration et de l'installation du PHP. Les connaissances requises sont les suivantes :

- Connaissances basiques d'UNIX (savoir faire un "make" et utiliser un compilateur C)
- Avoir un compilateur C ANSI installé
- Avoir installé un serveur web

Il y a plusieurs façons de compiler et configurer PHP pour une plate-forme UNIX. Le processus de compilation est contrôlé entièrement par les options de ligne de commande du script ``configure'` Cette documentation souligne les options les plus fréquentes, mais il en existe un grand nombre. Jetez un oeil à [4.3 Liste complète des options de configuration](#) pour une documentation exhaustive.

- [4.2.1 Installation rapide \(Version Module Apache\)](#)
- [4.2.2 Module fhttpd](#)
- Pour l'utiliser avec [4.2.3 Autres serveurs web](#)
- Sous forme d' [4.2.4 ``/usr/local/src/fhttpd'`](#)

4.2.1 Installation rapide (Version Module Apache)

[\[Notes en ligne\]](#)

PHP peut être compilé de nombreuses manières différentes pour en faire un module Apache. Voyons d'abord les instructions rapides, puis une liste d'exemples divers, avec les explications. Comme cela nous aurons survolé l'ensemble de l'installation.

Vous pouvez sélectionner les arguments à ajouter à la commande `configure` (ligne 8, ci dessous) parmi la liste [4.3 Liste complète des options de configuration](#).

Instructions d'installation rapide (Version Module Apache)

1. `gunzip apache_1.3.x.tar.gz`
2. `tar xvf apache_1.3.x.tar`
3. `gunzip php-3.0.x.tar.gz`
4. `tar xvf php-3.0.x.tar`


```

5. cd apache_1.3.x
6. ./configure --prefix=/www
7. cd ../php-3.0.x
8. ./configure --with-mysql --with-apache=../apache_1.3.x --enable-track-vars
9. make
10. make install
11. cd ../apache_1.3.x
12. ./configure --prefix=/www --activate-module=src/modules/php3/libphp3.a
13. make
14. make install

```

A la place de cette étape, vous pouvez simplement écraser le binaire httpd. Assurez-vous d'avoir bien arrêté le démon d'abord.

```

15. cd ../php-3.0.x
16. cp php3.ini-dist /usr/local/lib/php3.ini

```

Vous pouvez éditer le fichier de configuration /usr/local/lib/php3.ini. Si vous préférez installer le fichier dans un autre répertoire, il faut utiliser l'option de configuration `--with-config-file-path=/path` à l'étape 8.

17. Editez le fichier de configuration apache httpd.conf ou srm.conf et ajoutez :

```
AddType application/x-httpd-php3 .php3
```

Ici, il faut choisir l'extension que vous souhaitez donner au fichier php. .php est simplement celle que nous suggérons.

18. Utilisez la procédure normale afin de démarrer le serveur Apache. (Vous devez impérativement arrêter et redémarrer le serveur Apache, et pas seulement le relancer à l'aide d'un signal HUP ou USR1).

```
./configure --with-apxs --with-pgsql
```

Cette option va créer un fichier ``libphp4.so'`, qui est une librairie partagée, chargée par Apache grâce à la ligne `LoadModule` dans le fichier de configuration d'Apache ``httpd.conf'`. Le support de la base de données PostgreSQL est compris dans la librairie ``libphp4.so'`.

```
./configure --with-apxs --with-pgsql=shared
```

Cette option va créer un fichier ``libphp4.so'` qui est une librairie partagée Apache, mais elle va aussi créer une librairie partagée ``pgsql.so'` qui sera chargée par PHP soit avec les directives de chargement du fichier de configuration ``php.ini'` ou avec la fonction de chargement [dl\(\)](#).

```
./configure --with-apache=/path/to/apache_source --with-pgsql
```

Cette option va créer la librairie ``libmodphp4.a'`, le fichier ``mod_php4.c'` et quelques autres fichiers utilitaires, puis copier tout cela dans le dossier `src/modules/php4` du dossier source Apache. Lorsque vous compilez Apache avec l'option `--activate-module=src/modules/php4/libphp4.a`, le compilateur d'Apache créera le fichier ``libphp4.a'` et le liera statiquement avec ``httpd'`. Le support PostgreSQL est inclus directement dans l'exécutable ``httpd'`, ce qui fait que le résultat est un fichier exécutable unique ``httpd'`, qui combine Apache et tout PHP.

```
./configure --with-apache=/path/to/apache_source --with-pgsql=shared
```

Même chose que le précédent, mais au lieu d'inclure le support de PostgreSQL directement dans le ``httpd'` final, vous obtiendrez une librairie partagée ``pgsql.so'` que vous pourrez charger dans PHP, grâce au fichier de configuration ``php.ini'` ou la fonction [dl\(\)](#). Lors du choix de compilation de PHP, prenez bien en considération les avantages et les inconvénients de chaque méthode. Compiler PHP comme une librairie partagée vous permet de compiler séparément Apache et toutes les extensions PHP. Compiler PHP statiquement vous donne une plus grande vitesse d'exécution. Pour plus de détails reportez vous à la documentation Apache [support DSO](#) (en anglais).

4.2.2 Module fhttpd

[\[Notes en ligne\]](#)

Pour compiler PHP comme un module fhttpd, répondre "yes" à la question "Build as an fhttpd module ?" (cela correspond à l'option de configuration [4.3.7.6 install.configure.with-fhttpd=DIR](#) et spécifier la racine de la distribution fhttpd. Le répertoire par défaut est: ``/usr/local/src/fhttpd'`. Si vous utilisez fhttpd, compiler PHP en module vous permettra d'obtenir des performances supérieures, plus de contrôle et la possibilité d'exécution à distance.

4.2.3 Autres serveurs web

[\[Notes en ligne\]](#)

PHP peut être compilé pour fonctionner avec de nombreux autres serveurs web. Reportez vous à [4.3.7 Serveur](#) pour une liste complète des options de configuration.

4.2.4 ``/usr/local/src/fhttpd'`

[\[Notes en ligne\]](#)

Par défaut, PHP est compilé comme une CGI. Si vous voulez que votre serveur web supporte le PHP, compiler le PHP comme une CGI permet d'obtenir de meilleures performances. Cependant, la version CGI permet aux utilisateurs de lancer des scripts PHP sous leur UID respectives. Lisez attentivement le chapitre consacré à la [6 Sécurité](#) si vous souhaitez utiliser cette solution.

4.2.5 Options de base de données

[\[Notes en ligne\]](#)

PHP dispose du support natif d'un grand nombre de base de données et d'ODBC. Pour activer les différentes bases de données, vous pouvez utiliser les options du script ``configure'`, au moment de la compilation. Lisez la [4.3.1 Configuration pour le support des bases de données](#) pour plus de détails. La liste complète des options de ``configure'`, reportez vous à la [4.3 Liste complète des options de configuration](#).

4.2.6 Compilation

[\[Notes en ligne\]](#)

Lorsque PHP est configuré, vous êtes prêts à compiler un exécutable CGI, ou une librairie HP. La commande `make` devrait faire le travail. Si ce n'est pas le cas, et que vous comprenez pas pourquoi, allez à [4.5 Problèmes?](#).

4.2.7 Tests

[\[Notes en ligne\]](#)

Si vous avez compilé PHP comme programme CGI, vous pouvez tester votre produit en tapant : `make test`. C'est toujours une bonne chose de tester le résultat d'une compilation. Cela vous permet de repérer des problèmes entre PHP et votre plate-forme, bien plus facilement que si vous attendez.

4.2.8 Performances

[\[Notes en ligne\]](#)

Si vous avez compilé PHP comme programme CGI, vous pouvez évaluer les performances de PHP 3 avec la commande `make bench`. Notez que si le [7.1.3.1 ini.safe-mode](#) est activé (par défaut), vous ne risquez pas de voir l'évaluation s'arrêter une fois les 30 secondes réglementaires écoulées. En effet, la fonction `set_time_limit()` ne peut pas être utilisé si le [7.1.3.1 ini.safe-mode](#) fonctionne. Utilisez l'option [7.1.1.20 ini.max-execution-time](#) pour contrôler le temps d'exécutions de vos scripts. `make bench` ignore le fichier de [7.1 Le fichier de configuration](#).

Note : `make bench` n'est disponible qu'en PHP 3.

4.3 Liste complète des options de configuration

[\[Notes en ligne\]](#)

Note : *Ces options ne sont utilisées que lors de la compilation. Si vous voulez modifier le comportement de PHP au moment de l'exécution, reportez vous à la partie [7 Configuration](#).*

Ceci est la liste complète des options de configurations supportées par le script de configuration PHP 3 et PHP 4 ``configure``, utilisé lors de la compilation en environnement UNIX. Certaines sont disponibles sous PHP 3, d'autres sous PHP 4, d'autres encore pour les deux versions. De nombreuses options ont changé de nom entre PHP 3 et PHP 4, mais font exactement la même chose. Les entrées disposent d'un lien entre elles (si vous avez un problème avec une option PHP 3, vérifier ainsi que les noms n'ont pas changé).

- [4.3.1 Configuration pour le support des bases de données](#)
- [4.3.2 Ecommerce](#)
- [4.3.3 Graphisme](#)
- [4.3.4 Divers](#)
- [4.3.5 Réseau](#)
- [4.3.6 Configuration de PHP](#)
- [4.3.7 Serveur](#)
- [4.3.8 Texte et langue](#)
- [4.3.9 XML](#)

4.3.1 Configuration pour le support des bases de données

[\[Notes en ligne\]](#)

PHP supporte de nombreuses bases de données (et aussi ODBC):

Pour la liste détaillée des options de ``configure``, reportez vous à [4.3 Liste complète des options de configuration](#).

[4.3.1.1 install.configure.with-adabas](#)

[\[Notes en ligne\]](#)

--with-adabas[=DIR]

- PHP 3, PHP 4: Ajoute le support Adabas. DIR est le dossier de l'installation Adabas (par défaut : /usr/local).

[Site d'Adabas](#)

[4.3.1.2 install.configure.enable-dba](#)

[\[Notes en ligne\]](#)

--enable-dba=shared

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile DBA comme librairie partagée

[4.3.1.3 install.configure.enable-dbase](#)

[\[Notes en ligne\]](#)

--enable-dbase

- PHP 3: Option non disponible; utilisez [4.3.1.4 install.configure.with-dbase](#).
PHP 4: Active la librairie dbase. Par de librairie externe nécessaire.

[4.3.1.4 install.configure.with-dbase](#)

[\[Notes en ligne\]](#)

--with-dbase

- PHP 3: Inclus la librairie dbase. Par de librairie externe nécessaire.
PHP 4: Option non disponible; utilisez [4.3.1.3 install.configure.enable-dbase](#).

[4.3.1.5 install.configure.with-db2](#)

[\[Notes en ligne\]](#)

--with-db2[=DIR]

- PHP 3, PHP 4: Ajoute le support de Berkeley DB2

[4.3.1.6 install.configure.with-db3](#)

[\[Notes en ligne\]](#)

--with-db3[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Berkeley DB3

[4.3.1.7 install.configure.with-dbm](#)

[\[Notes en ligne\]](#)

--with-dbm[=DIR]

- PHP 3, PHP 4: Inclus le support DBM

[4.3.1.8 install.configure.with-dbmaker](#)

[\[Notes en ligne\]](#)

--with-dbmaker[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support DBMaker. DIR est le dossier de l'installation de DBMaker (par défaut, c'est le chemin jusqu'à la dernière installation de DBMaker, ou /home/dbmaker/3.6).

[4.3.1.9 install.configure.with-empress](#)

[\[Notes en ligne\]](#)

--with-empress[=DIR]

- PHP 3, PHP 4: Inclus le support Empress. DIR est le dossier d'installation d'Empress (par défaut, \$EMPRESSPATH).

[4.3.1.10 install.configure.enable-filepro](#)

[\[Notes en ligne\]](#)

--enable-filepro

- PHP 3: Option non disponible; utilisez [4.3.1.11 install.configure.with-filepro](#).
PHP 4: Active le support de filePro (lecture seule). Par de librairie externe nécessaire.

[4.3.1.11 install.configure.with-filepro](#)

[\[Notes en ligne\]](#)

--with-filepro

- PHP 3: Active le support de filePro (lecture seule). Par de librairie externe nécessaire.
PHP 4: Option non disponible; utilisez [4.3.1.10 install.configure.enable-filepro](#).

[4.3.1.12 install.configure.with-gdbm](#)

[\[Notes en ligne\]](#)

--with-gdbm[=DIR]

- PHP 3, PHP 4: Inclus le support GDBM

[4.3.1.13 install.configure.with-hyperwave](#)

[\[Notes en ligne\]](#)

--with-hyperwave

- PHP 3, PHP 4: Inclus le support Hyperwave

[4.3.1.14 install.configure.with-ibm-db2](#)

[\[Notes en ligne\]](#)

`--with-ibm-db2[=DIR]`

- PHP 3, PHP 4: Inclus le support IBM DB2. DIR est le chemin jusqu'au dossier d'installation de DB2 (par défaut ``/home/db2inst1/sqllib'`).

[Site IBM DB2](#)

[4.3.1.15 install.configure.with-informix](#)

[\[Notes en ligne\]](#)

`--with-informix[=DIR]`

- PHP 3, PHP 4: Inclus le support Informix. DIR est le dossier d'installation de Informix (pas de valeur par défaut).

[4.3.1.16 install.configure.with-ingres](#)

[\[Notes en ligne\]](#)

`--with-ingres[=DIR]`

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Ingres II support. DIR est le dossier d'installation de Ingres (par défaut `/II/ingres`)

[4.3.1.17 install.configure.with-interbase](#)

[\[Notes en ligne\]](#)

`--with-interbase[=DIR]`

- PHP 3, PHP 4: Inclus le support InterBase. DIR est le dossier d'installation de InterBase (par défaut ``/usr/interbase'`).

[10.29 Fonctions InterBase](#)

[Site Interbase](#)

[4.3.1.18 install.configure.with-ldap](#)

[\[Notes en ligne\]](#)

`--with-ldap[=DIR]`

- PHP 3: Inclus le support LDAP. DIR est le dossier d'installation de LDAP (par défaut ``/usr'` et ``/usr/local'`).
- PHP 4: Inclus le support LDAP. DIR est le dossier d'installation de LDAP.
Cette option fournit un accès aux serveurs **LDAP** (Lightweight Directory Access Protocol support).
Le paramètre est le dossier d'installation de LDAP (par défaut, ``/usr/local/ldap'`).
Pour plus de détails sur LDAP allez à [RFC1777](#) et [RFC1778](#).

[4.3.1.19 install.configure.with-msql](#)

[\[Notes en ligne\]](#)

`--with-msql[=DIR]`

- PHP 3, PHP 4: Active le support mSQL. Le paramètre est le dossier d'installation de mSQL (par défaut, ``/usr/local/Hughes'`, le dossier par défaut de l'installation mSQL 2.0). configure automatiquement détecte quelle version de mSQL vous utilisez et gère le point avec les deux versions 1.0 et 2.0, mais si vous compilez PHP avec mSQL 1.0, vous ne pourrez accéder qu'aux bases mSQL 1.0, et vice-versa.
Voir aussi les directives de [7.1.7 Directives de configuration mSQL](#) du fichier [7.1 Le fichier de configuration](#).
[site mSQL](#)

[4.3.1.20 install.configure.with-mysql](#)

[\[Notes en ligne\]](#)

`--with-mysql[=DIR]`

- PHP 3: Inclus le support MySQL. DIR est le dossier d'installation de MySQL (par défaut, il recherche dans les lieux standards d'installation de MySQL).
PHP 4: Inclus le support MySQL. DIR est le dossier d'installation de MySQL. Si il n'est pas spécifié, la librairie MySQL sera utilisée (option par défaut).
Voir aussi les directives de [7.1.6 MySQL Configuration Directives](#) du fichier de [7.1 Le fichier de configuration](#).
[Site de MySQL](#)

[4.3.1.21 install.configure.with-ndbm](#)

[\[Notes en ligne\]](#)

`--with-ndbm[=DIR]`

- PHP 3, PHP 4: Inclus le support NDBM

[4.3.1.22 install.configure.with-oci8](#)

[\[Notes en ligne\]](#)

`--with-oci8[=DIR]`

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Oracle-oci8. DIR est le dossier d'installation de Oracle (par défaut ORACLE_HOME).

[4.3.1.23 install.configure.with-oracle](#)

[\[Notes en ligne\]](#)

`--with-oracle[=DIR]`

- PHP 3: Inclus le support Oracle . DIR est le dossier d'installation de Oracle (par défaut \$ORACLE_HOME).
PHP 4: Inclus le support Oracle-oci7. DIR est le dossier d'installation de Oracle (par défaut

ORACLE_HOME).

Inclus le support Oracle. Cette option a été testée et devrait fonctionner au moins avec les versions d'Oracle de 7.0 à 7.3. Le paramètre DIR est *ORACLE_HOME*. Vous n'avez pas à spécifier ce paramètre, si votre environnement Oracle a été installé.

[Site Oracle](#)

[4.3.1.24 install.configure.with-pgsql](#)

[\[Notes en ligne\]](#)

--with-pgsql[=DIR]

- PHP 3: Inclus le support PostgreSQL. DIR est le dossier d'installation de PostgreSQL (par défaut ``/usr/local/pgsql'``).
 - PHP 4: Inclus le support PostgreSQL. DIR est le dossier d'installation de PostgreSQL (par défaut ``/usr/local/pgsql'``). Utilisez "shared" pour compiler PostgreSQL en librairie dynamique, ou "shared,DIR" pour compiler PostgreSQL en librairie dynamique tout en spécifiant DIR.
- Voir aussi les directives de configurations de [7.1.8 Directives de configuration Postgres](#) dans le fichier de [7.1 Le fichier de configuration](#).

[Site PostgreSQL](#)

[4.3.1.25 install.configure.with-solid](#)

[\[Notes en ligne\]](#)

--with-solid[=DIR]

- PHP 3, PHP 4: Inclus le support Solid. DIR est le dossier d'installation de Solid (par défaut `/usr/local/solid`).

[Site Solid](#)

[4.3.1.26 install.configure.with-sybase-ct](#)

[\[Notes en ligne\]](#)

--with-sybase-ct[=DIR]

- PHP 3, PHP 4: Inclus le support Sybase-CT. DIR est le dossier d'installation de Sybase (par défaut `/home/sybase`).
- Voir aussi les directives de configuration de [7.1.11 Sybase-CT Configuration Directives](#) dans le fichier de [7.1 Le fichier de configuration](#).

[4.3.1.27 install.configure.with-sybase](#)

[\[Notes en ligne\]](#)

--with-sybase[=DIR]

- PHP 3, PHP 4: Inclus le support Sybase-DB. DIR est le dossier d'installation de Sybase (par défaut ``/home/sybase'``).
- Voir aussi les directives de configuration de [7.1.10 Directives de configuration Sybase](#) dans le dossier de [7.1 Le fichier de configuration](#).

[site Sybase](#)

[4.3.1.28 install.configure.with-openlink](#)

[\[Notes en ligne\]](#)

`--with-openlink[=DIR]`

- PHP 3, PHP 4: Inclus le support OpenLink ODBC . DIR est le dossier d'installation de OpenLink (par défaut /usr/local/openlink).

[Site OpenLink Software](#)

[4.3.1.29 install.configure.with-iodbc](#)

[\[Notes en ligne\]](#)

`--with-iodbc[=DIR]`

- PHP 3, PHP 4: Inclus le support iODBC. DIR est le dossier d'installation de iODBC (par défaut '/usr/local').

Cette option a été développée pour le gestionnaire de pilote iODBC, qui était librement redistribuable, et fonctionnait sous la plus part des UNIX.

[Site FreeODBC](#) or [Site iODBC](#)

[4.3.1.30 install.configure.with-custom-odbc](#)

[\[Notes en ligne\]](#)

`--with-custom-odbc[=DIR]`

- PHP 3, PHP 4: Inclus le support pour une librairie arbitraire ODBC. Le paramètre est le dossier d'installation et par défaut, c'est '/usr/local'.
- Cette option implique que vous avez défini la variable CUSTOM_ODBC_LIBS, lorsque vous exécutez le script de configuration. Vous devez aussi fournir un entête odbc.h valide, dans le chemin d'inclusion. Si vous n'en avez pas, créez le, et incluez y vos entêtes spécifiques. Ces entêtes peuvent aussi utiliser des définitions supplémentaires, surtout si ils sont multi-plateformes. Définissez les dans CFLAGS.

Par exemple, vous pouvez utiliser Sybase SQL Anywhere sous QNX comme suit :

```
CFLAGS=-DODBC_QNX LDFLAGS=-lnix CUSTOM_ODBC_LIBS="-ldblib -lodbc"
./configure --with-custom-odbc=/usr/lib/sqlany50
```

[4.3.1.31 install.configure.disable-unified-odbc](#)

[\[Notes en ligne\]](#)

`--disable-unified-odbc`

- PHP 3: Inactive le support ODBC unifié. Uniquement utile si iODBC, Adabas, Solid, Velocis ou une interface ODBC personnalisée est activée.

PHP 4: Option non disponible en PHP 4

Le module ODBC unifié, qui est une interface répandue vers toutes les bases de données ODBC, comme par exemple Solid, IBM DB2 et Adabas D. Elle fonctionne aussi pour les librairie ODBC normales. Elle a été testée avec iODBC, Solid, Adabas D, IBM DB2 et Sybase SQL Anywhere. Requiert qu'un (et seulement un) de ces module ait été activé, ou que le module Velocis soit activé, ou encore que la librairie ODBC personnalisée soit chargée. Cette option ne sert que si l'une de ces options est utilisée : [4.3.1.29 install.configure.with-iodbc](#), [4.3.1.25 install.configure.with-solid](#), [4.3.1.14 install.configure.with-ibm-db2](#), [4.3.1.1 install.configure.with-adabas](#), [4.3.1.33](#)

[install.configure.with-velocis](#), ou [4.3.1.30 install.configure.with-custom-odbc](#).

Voir aussi les directives de configuration de [7.1.15 Directives de configuration du driver ODBC unifié](#) dans le fichier de [7.1 Le fichier de configuration](#).

[4.3.1.32 install.configure.with-unixODBC](#)

[\[Notes en ligne\]](#)

--with-unixODBC[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support unixODBC. DIR est le dossier d'installation de unixODBC (par défaut /usr/local).

[4.3.1.33 install.configure.with-velocis](#)

[\[Notes en ligne\]](#)

--with-velocis[=DIR]

- PHP 3, PHP 4: Inclus le support Velocis. DIR est le dossier d'installation de Velocis (par défaut /usr/local/velocis).
[Site Velocis](#)

[4.3.2 Ecommerce](#)

[\[Notes en ligne\]](#)

[4.3.2.1 install.configure.with-ccvs](#)

[\[Notes en ligne\]](#)

--with-ccvs[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support CCVS de PHP 4. Passez le dossier d'installation de CCVS comme valeur de DIR.

[4.3.2.2 install.configure.with-mck](#)

[\[Notes en ligne\]](#)

--with-mck[=DIR]

- PHP 3: Inclus le support Cybercash MCK. DIR est le dossier d'installation de cybercash mck (par défaut ``/usr/src/mck-3.2.0.3-linux'`). Pour plus d'informations, ``extra/cyberlib'`.
PHP 4: Option non disponible; utilisez [4.3.2.3 install.configure.with-cybercash](#).

[4.3.2.3 install.configure.with-cybercash](#)

[\[Notes en ligne\]](#)

--with-cybercash[=DIR]

- PHP 3: Option non disponible; utilisez plutôt [4.3.2.2 install.configure.with-mck](#).
PHP 4: Inclus le support CyberCash. DIR est le dossier d'installation de CyberCash MCK.

[4.3.2.4 install.configure.with-pfpro](#)

[\[Notes en ligne\]](#)

--with-pfpro[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus de support Verisign Payflow Pro.

[4.3.3 Graphisme](#)

[\[Notes en ligne\]](#)

[4.3.3.1 install.configure.enable-freetype-4bit-antialias-hack](#)

[\[Notes en ligne\]](#)

--enable-freetype-4bit-antialias-hack

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support de FreeType2 (expérimental).

[4.3.3.2 install.configure.with-gd](#)

[\[Notes en ligne\]](#)

--with-gd[=DIR]

- PHP 3: Inclus le support GD (DIR est le dossier d'installation de GD).
PHP 4: Inclus le support GD (DIR est le dossier d'installation de GD). Utilisez "shared" pour compiler GD en librairie dynamique, ou "shared,DIR" pour compiler GD en librairie dynamique tout en spécifiant DIR.

[4.3.3.3 install.configure.without-gd](#)

[\[Notes en ligne\]](#)

--without-gd

- PHP 3, PHP 4: Inactive le support GD.

[4.3.3.4 install.configure.with-imagick](#)

[\[Notes en ligne\]](#)

--with-imagick[=DIR]

- PHP 3: Inclus le support ImageMagick. DIR est le dossier d'installation (par défaut, PHP essaiera de se débrouiller).[expérimental]
PHP 4: Option non disponible en PHP 4

[4.3.3.5 install.configure.with-jpeg-dir](#)

[\[Notes en ligne\]](#)

--with-jpeg-dir[=DIR]

- PHP 3: Dossier jpeg pour pdflib 2.0
PHP 4: Dossier jpeg pour pdflib 3.x

[4.3.3.6 install.configure.with-png-dir](#)

[\[Notes en ligne\]](#)

--with-png-dir[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Dossier png pour pdflib 3.x

[4.3.3.7 install.configure.enable-t1lib](#)

[\[Notes en ligne\]](#)

--enable-t1lib

- PHP 3: Active le support t1lib.
PHP 4: Option non disponible; utilisez plutôt [4.3.3.8 install.configure.with-t1lib](#).

[4.3.3.8 install.configure.with-t1lib](#)

[\[Notes en ligne\]](#)

--with-t1lib[=DIR]

- PHP 3: Option non disponible; utilisez plutôt [4.3.3.7 install.configure.enable-t1lib](#).
PHP 4: Inclus le support t1lib.

[4.3.3.9 install.configure.with-tiff-dir](#)

[\[Notes en ligne\]](#)

--with-tiff-dir[=DIR]

- PHP 3: Dossier tiff pour pdflib 2.0
PHP 4: Dossier tiff pour pdflib 3.x

[4.3.3.10 install.configure.with-ttf](#)

[\[Notes en ligne\]](#)

--with-ttf[=DIR]

- PHP 3, PHP 4: Inclus le support ttf

[4.3.3.11 install.configure.with-xpm-dir](#)

[\[Notes en ligne\]](#)

--with-xpm-dir[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Dossier xpm pour gd-1.8+

4.3.4 Divers

[\[Notes en ligne\]](#)

Cette section est en cours de tri.

[4.3.4.1 install.configure.disable-bcmath](#)

[\[Notes en ligne\]](#)

--disable-bcmath

- PHP 3: Inactive la librairie de nombre BC (taille arbitraire). Ces fonctions vous permettent de manipuler des nombres hors des limites habituelles des entiers et des nombres à virgules flottantes. Voir aussi [10.4 Nombres de grande taille](#).
PHP 4: Option non disponible; bcmath n'est pas compilé par défaut. Utilisez [4.3.4.9 install.configure.enable-bcmath](#).

[4.3.4.2 install.configure.disable-display-source](#)

[\[Notes en ligne\]](#)

--disable-display-source

- PHP 3: Compilation sans affichage du source.
PHP 4: Option non disponible en PHP 4

[4.3.4.3 install.configure.disable-libtool-lock](#)

[\[Notes en ligne\]](#)

--disable-libtool-lock

- PHP 3: Option non disponible en PHP 3
PHP 4: Empêche le verrouillage (peut bloquer certaines compilations parallèle).

[4.3.4.4 install.configure.disable-pear](#)

[\[Notes en ligne\]](#)

--disable-pear

- PHP 3: Option non disponible en PHP 3
PHP 4: N'installe pas PEAR

[4.3.4.5 install.configure.disable-pic](#)

[\[Notes en ligne\]](#)

--disable-pic

- PHP 3: Option non disponible en PHP 3
PHP 4: Inactive PIC pour les objets partagés.

[4.3.4.6 install.configure.disable-posix](#)

[\[Notes en ligne\]](#)

--disable-posix

- PHP 3: Option non disponible en PHP 3; utilisez [4.3.4.61 install.configure.without-posix](#).
PHP 4: Désactive les fonctions POSIX.

[4.3.4.7 install.configure.disable-rpath](#)

[\[Notes en ligne\]](#)

--disable-rpath

- PHP 3: Option non disponible en PHP 3
PHP 4: Inactive le passage automatique de dossier de recherche supplémentaires.

[4.3.4.8 install.configure.disable-session](#)

[\[Notes en ligne\]](#)

--disable-session

- PHP 3: Option non disponible en PHP 3
PHP 4: Inactive les sessions

[4.3.4.9 install.configure.enable-bcmath](#)

[\[Notes en ligne\]](#)

--enable-bcmath

- PHP 3: Option non disponible en PHP 3; bcmath est compilé par Utilisez [4.3.4.1 install.configure.disable-bcmath](#) pour le désactiver.
PHP 4: Compile PHP avec les fonctions mathématiques BC. Lisez le fichier README-BCMATH pour connaître les instructions d'installation de ce module. Ces fonctions vous permettent de travailler avec des nombres de tailles illimitées. Reportez vous aux fonctions [10.4 Nombres de grande taille](#) pour plus d'informations.

[4.3.4.10 install.configure.enable-c9x-inline](#)

[\[Notes en ligne\]](#)

--enable-c9x-inline

- PHP 3: Option non disponible en PHP 3
PHP 4: Active la sémantique C9x-inline.

[4.3.4.11 install.configure.enable-calendar](#)

[\[Notes en ligne\]](#)

--enable-calendar

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support des conversions calendaires

[4.3.4.12 install.configure.enable-debug](#)

[\[Notes en ligne\]](#)

--enable-debug

- PHP 3, PHP 4: Ajoute le débogage des symboles.

[4.3.4.13 install.configure.enable-debugger](#)

[\[Notes en ligne\]](#)

--enable-debugger

- PHP 3: Ajoute les fonctions de débogage distants.
PHP 4: Option non disponible en PHP 4

[4.3.4.14 install.configure.enable-discard-path](#)

[\[Notes en ligne\]](#)

--enable-discard-path

- PHP 3, PHP 4: Si cette option est activée, l'exécutable PHP CGI peut être placé sûrement hors de l'arborescence web, et personne ne pourra contourner la sécurité .htaccess.

[4.3.4.15 install.configure.enable-dmalloc](#)

[\[Notes en ligne\]](#)

--enable-dmalloc

- PHP 3, PHP 4: Active dmalloc

[4.3.4.16 install.configure.enable-exif](#)

[\[Notes en ligne\]](#)

--enable-exif

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support exif

[4.3.4.17 install.configure.enable-experimental-zts](#)

[\[Notes en ligne\]](#)

--enable-experimental-zts

- PHP 3: Option non disponible en PHP 3
PHP 4: Cette option risque de détruire votre compilation

[4.3.4.18 install.configure.enable-fast-install](#)

[\[Notes en ligne\]](#)

--enable-fast-install[=PKGS]

- PHP 3: Option non disponible en PHP 3
PHP 4: Optimise la vitesse d'exécution [par défaut=yes]

[4.3.4.19 install.configure.enable-force-cgi-redirect](#)

[\[Notes en ligne\]](#)

--enable-force-cgi-redirect

- PHP 3, PHP 4: Active la vérification des redirections internes au serveur. Il est recommandé d'utiliser cette option si vous utilisez la version CGI avec Apache.

[4.3.4.20 install.configure.enable-inline-optimization](#)

[\[Notes en ligne\]](#)

--enable-inline-optimization

- PHP 3: Option non disponible en PHP 3
PHP 4: Si vous avez beaucoup de mémoire libre, et que vous utilisez gcc, vous pouvez essayer cela.

[4.3.4.21 install.configure.enable-libgcc](#)

[\[Notes en ligne\]](#)

--enable-libgcc

- PHP 3: Option non disponible en PHP 3
PHP 4: Active explicitement le linkage avec libgcc

[4.3.4.22 install.configure.enable-maintainer-mode](#)

[\[Notes en ligne\]](#)

--enable-maintainer-mode

- PHP 3, PHP 4: Active des règles de compilation rarement utile (et même parfois confuse) à l'installateur occasionnel.

[4.3.4.23 install.configure.enable-memory-limit](#)

[\[Notes en ligne\]](#)

--enable-memory-limit

- PHP 3, PHP 4: Compile avec limitation de consommation de mémoire. [default=no]

[4.3.4.24 install.configure.enable-safe-mode](#)

[\[Notes en ligne\]](#)

--enable-safe-mode

- PHP 3, PHP 4: Active le [7.1.3.1 ini.safe-mode](#) par défaut.

[4.3.4.25 install.configure.enable-satellite](#)

[\[Notes en ligne\]](#)

--enable-satellite

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support CORBA via Satellite (Requiert ORBit)

[4.3.4.26 install.configure.enable-shared](#)

[\[Notes en ligne\]](#)

--enable-shared[=PKGS]

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile les bibliothèques partagées [par défaut=yes]

[4.3.4.27 install.configure.enable-sigchild](#)

[\[Notes en ligne\]](#)

--enable-sigchild

- PHP 3, PHP 4: Active le gestionnaire SIGCHLD interne à PHP.

[4.3.4.28 install.configure.enable-static](#)

[\[Notes en ligne\]](#)

--enable-static[=PKGS]

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile les bibliothèques statiquement [par défaut=yes]

[4.3.4.29 install.configure.enable-sysvsem](#)

[\[Notes en ligne\]](#)

--enable-sysvsem

- PHP 3, PHP 4: Active le support des sémaphores System V.

[4.3.4.30 install.configure.enable-sysvshm](#)

[\[Notes en ligne\]](#)

--enable-sysvshm

- PHP 3, PHP 4: Active le système System V de mémoire partagée

[4.3.4.31 install.configure.enable-trans-sid](#)

[\[Notes en ligne\]](#)

--enable-trans-sid

- PHP 3: Option non disponible en PHP 3

PHP 4: Active la propagation transparente d'identification de session

[4.3.4.32 install.configure.with-cdb](#)

[\[Notes en ligne\]](#)

--with-cdb[=DIR]

- PHP 3, PHP 4: Inclus le support cdb

[4.3.4.33 install.configure.with-config-file-path](#)

[\[Notes en ligne\]](#)

--with-config-file-path=PATH

- PHP 3: Indique le chemin dans lequel il faut chercher le fichier ``php3.ini'` (par défaut, ``/usr/local/lib'`).
- PHP 4: Indique le chemin dans lequel il faut chercher le fichier ``php.ini'` (par défaut, ``/usr/local/lib'`).

[4.3.4.34 install.configure.with-cpdflib](#)

[\[Notes en ligne\]](#)

--with-cpdflib[=DIR]

- PHP 3: Inclus le support cpdflib. DIR est le dossier d'installation de ClibPDF (par défaut, `/usr/local`).
- PHP 4: Inclus le support cpdflib (requiert cpdflib >= 2). DIR est le dossier d'installation de ClibPDF (par défaut, `/usr/local`).

[4.3.4.35 install.configure.with-esoob](#)

[\[Notes en ligne\]](#)

--with-esoob[=DIR]

- PHP 3: Option non disponible en PHP 3
- PHP 4: Inclus le support Easysoft OOB. DIR est le dossier d'installation de OOB (par défaut `/usr/local/easysoft/oob/client`).

[4.3.4.36 install.configure.with-exec-dir](#)

[\[Notes en ligne\]](#)

--with-exec-dir[=DIR]

- PHP 3, PHP 4: Autorise uniquement les exécutables dans le dossier DIR lorsque le [7.1.3.1 ini.safe-mode](#) est activé (par défaut, `/usr/local/php/bin`).

[4.3.4.37 install.configure.with-fdftk](#)

[\[Notes en ligne\]](#)

--with-fdftk[=DIR]

- PHP 3, PHP 4: Inclus le support fdftk. DIR est le dossier d'installation de fdftk (par défaut, `/usr/local`).

[4.3.4.38 install.configure.with-gnu-ld](#)

[\[Notes en ligne\]](#)

`--with-gnu-ld`

- PHP 3: Option non disponible en PHP 3
PHP 4: Support que le compilateur C utilise GNU ld [par défaut =no]

[4.3.4.39 install.configure.with-icap](#)

[\[Notes en ligne\]](#)

`--with-icap[=DIR]`

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support ICAP.

[4.3.4.40 install.configure.with-imap](#)

[\[Notes en ligne\]](#)

`--with-imap[=DIR]`

- PHP 3, PHP 4: Inclus le support **IMAP**. DIR est le dossier d'installation d'**IMAP** (celui qui contient les dossier include et c-client.a).

[4.3.4.41 install.configure.with-imsf](#)

[\[Notes en ligne\]](#)

`--with-imsf[=DIR]`

- PHP 3: Inclus le support **IMSP** (DIR est le dossier d'installation d'**IMSP** (celui qui contient les dossiers include et c-client.a).
PHP 4: Option non disponible en PHP 4

[4.3.4.42 install.configure.with-java](#)

[\[Notes en ligne\]](#)

`--with-java[=DIR]`

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Java. DIR est le dossier d'installation du JDK. Cette extension ne peut être compilé que comme une librairie partagée.

[4.3.4.43 install.configure.with-kerberos](#)

[\[Notes en ligne\]](#)

`--with-kerberos[=DIR]`

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Kerberos avec **IMAP**.

[4.3.4.44 install.configure.with-mcal](#)

[\[Notes en ligne\]](#)

--with-mcal[=DIR]

- PHP 3, PHP 4: Inclus le support MCAL.

[4.3.4.45 install.configure.with-mcrypt](#)

[\[Notes en ligne\]](#)

--with-mcrypt[=DIR]

- PHP 3, PHP 4: Inclus le support mcrypt. DIR est le dossier d'installation de mcrypt.

[4.3.4.46 install.configure.with-mhash](#)

[\[Notes en ligne\]](#)

--with-mhash[=DIR]

- PHP 3, PHP 4: Inclus le support mhash. DIR est le dossier d'installation de mhash.

[4.3.4.47 install.configure.with-mm](#)

[\[Notes en ligne\]](#)

--with-mm[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support mm pour le stockage des sessions

[4.3.4.48 install.configure.with-mod_charset](#)

[\[Notes en ligne\]](#)

--with-mod_charset

- PHP 3, PHP 4: Active les tables de transfert pour mod_charset (Rus Apache).

[4.3.4.49 install.configure.with-pdflib](#)

[\[Notes en ligne\]](#)

--with-pdflib[=DIR]

- PHP 3: Inclus le support pdflib (testée avec 0.6 et 2.0). DIR est le dossier d'installation de pdflib (par défaut, ``/usr/local'`).
- PHP 4: Inclus le support pdflib 3.x. DIR est le dossier d'installation de pdflib (par défaut, ``/usr/local'`).

[4.3.4.50 install.configure.with-readline](#)

[\[Notes en ligne\]](#)

--with-readline[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support readline. DIR est le dossier d'installation de readline.

[4.3.4.51 install.configure.with-regex](#)

[\[Notes en ligne\]](#)

--with-regex=TYPE

- PHP 3: Option non disponible en PHP 3
PHP 4: Librairie de regex : system, apache, php

[4.3.4.52 install.configure.with-servlet](#)

[\[Notes en ligne\]](#)

--with-servlet[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support des servlets . DIR est le dossier d'installation du JSDK. Ces SAPI requiert la compilation de l'extension Java comme librairie partagée.

[4.3.4.53 install.configure.with-swf](#)

[\[Notes en ligne\]](#)

--with-swf[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support swf.

[4.3.4.54 install.configure.with-system-regex](#)

[\[Notes en ligne\]](#)

--with-system-regex

- PHP 3: N'utilise pas la librairie de regex fournie.
PHP 4: (obsolète) Utilise la librairie de regex.

[4.3.4.55 install.configure.with-tsrm-pth](#)

[\[Notes en ligne\]](#)

--with-tsrm-pth[=pth-config]

- PHP 3: Option non disponible en PHP 3
PHP 4: Utilise GNU Pth.

[4.3.4.56 install.configure.with-tsrm-pthreads](#)

[\[Notes en ligne\]](#)

--with-tsrm-pthreads

- PHP 3: Option non disponible en PHP 3
PHP 4: Utilise les threads POSIX (par défaut)

[4.3.4.57 install.configure.with-x](#)

[\[Notes en ligne\]](#)

`--with-x`

- PHP 3: Utilise le système X Window
PHP 4: Option non disponible en PHP 4

[4.3.4.58 install.configure.with-zlib-dir](#)

[\[Notes en ligne\]](#)

`--with-zlib-dir[=DIR]`

- PHP 3: Dossier zlib pour pdflib 2.0 ou Inclus le support zlib
PHP 4: Dossier zlib pour pdflib 3.x ou Inclus le support zlib

[4.3.4.59 install.configure.with-zlib](#)

[\[Notes en ligne\]](#)

`--with-zlib[=DIR]`

- PHP 3, PHP 4: Inclus le support zlib (requiert zlib >= 1.0.9). DIR est le dossier d'installation de zlib (par défaut, /usr).

[4.3.4.60 install.configure.without-pcre-regex](#)

[\[Notes en ligne\]](#)

`--without-pcre-regex`

- PHP 3: Exclut les expressions régulières compatibles Perl.
PHP 4: Exclut les expressions régulières compatibles Perl. Utilisez `--with-pcre-regex=DIR` pour spécifier le dossier DIR qui contient les dossier d'include et de libraries, si vous n'utilisez pas la librairie fournie.

[4.3.4.61 install.configure.without-posix](#)

[\[Notes en ligne\]](#)

`--without-posix`

- PHP 3: Exclut les fonctions POSIX
PHP 4: Option non disponible en PHP 4; utilisez [4.3.4.6 install.configure.disable-posix](#).

[4.3.5 Réseau](#)

[\[Notes en ligne\]](#)

[4.3.5.1 install.configure.with-curl](#)

[\[Notes en ligne\]](#)

`--with-curl[=DIR]`

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support CURL

[4.3.5.2 install.configure.enable-ftp](#)

[\[Notes en ligne\]](#)

--enable-ftp

- PHP 3: Option non disponible; utilisez plutôt [4.3.5.3 install.configure.with-ftp](#).
PHP 4: Enable FTP support

[4.3.5.3 install.configure.with-ftp](#)

[\[Notes en ligne\]](#)

--with-ftp

- PHP 3: Inclus le support FTP.
PHP 4: Option non disponible; utilisez plutôt [4.3.5.2 install.configure.enable-ftp](#) instead

[4.3.5.4 install.configure.disable-url-fopen-wrapper](#)

[\[Notes en ligne\]](#)

--disable-url-fopen-wrapper

- PHP 3, PHP 4: Inactive la version améliorée de fopen, qui permet l'accès aux fichiers par **HTTP** ou **FTP**.

[4.3.5.5 install.configure.with-mod-dav](#)

[\[Notes en ligne\]](#)

--with-mod-dav=DIR

- PHP 3, PHP 4: Inclus le support DAV grâce à mod_dav, DIR est le dossier d'installation de mod_dav's (version Apache module seulement!)

[4.3.5.6 install.configure.with-openssl](#)

[\[Notes en ligne\]](#)

--with-openssl[=DIR]

- PHP 3, PHP 4: Inclus le support OpenSSL de **SNMP**.

[4.3.5.7 install.configure.with-snmpp](#)

[\[Notes en ligne\]](#)

--with-snmpp[=DIR]

- PHP 3, PHP 4: Inclus le support **SNMP**. DIR est le dossier d'installation de **SNMP** (par défaut, PHP va chercher dans les un certain nombre d'endroits classiques d'installation de **SNMP**). Utilisez "shared" pour compiler **SNMP** en librairie dynamique, ou "shared,DIR" pour compiler **SNMP** en librairie dynamique tout en spécifiant DIR.

[4.3.5.8 install.configure.enable-ucd-snmp-hack](#)

[\[Notes en ligne\]](#)

--enable-ucd-snmp-hack

- PHP 3, PHP 4: Active le hack UCD *SNMP*

[4.3.5.9 install.configure.enable-sockets](#)

[\[Notes en ligne\]](#)

--enable-sockets

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support des sockets

[4.3.5.10 install.configure.with-yaz](#)

[\[Notes en ligne\]](#)

--with-yaz[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support YAZ (ANSI/NISO Z39.50). DIR est le dossier d'installation de YAZ.

[4.3.5.11 install.configure.enable-yp](#)

[\[Notes en ligne\]](#)

--enable-yp

- PHP 3: Option non disponible; utilisez plutôt [4.3.5.12 install.configure.with-yp](#).
PHP 4: Inclus le support YP

[4.3.5.12 install.configure.with-yp](#)

[\[Notes en ligne\]](#)

--with-yp

- PHP 3: Inclus le support YP
PHP 4: Option non disponible; utilisez plutôt [4.3.5.11 install.configure.enable-yp](#).

[4.3.6 Configuration de PHP](#)

[\[Notes en ligne\]](#)

[4.3.6.1 install.configure.enable-magic-quotes](#)

[\[Notes en ligne\]](#)

--enable-magic-quotes

- PHP 3, PHP 4: Active les magic quotes par défaut.

[4.3.6.2 install.configure.disable-short-tags](#)

[\[Notes en ligne\]](#)

--disable-short-tags

- PHP 3, PHP 4: Désactive les balises courtes (<?).

[4.3.6.3 install.configure.enable-track-vars](#)

[\[Notes en ligne\]](#)

--enable-track-vars

- PHP 3: Active la lecture des variables GET/POST/Cookie par défaut.
PHP 4: Option non disponible en PHP 4; sous PHP 4.0.2, track_vars est toujours à on.

[4.3.7 Serveur](#)

[\[Notes en ligne\]](#)

[4.3.7.1 install.configure.with-aolserver-src](#)

[\[Notes en ligne\]](#)

--with-aolserver-src=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Specifie le chemin jusqu'à la distribution du serveur AOLserver

[4.3.7.2 install.configure.with-aolserver](#)

[\[Notes en ligne\]](#)

--with-aolserver=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Specifie le chemin jusqu'au serveur AOLserver

[4.3.7.3 install.configure.with-apache](#)

[\[Notes en ligne\]](#)

--with-apache[=DIR]

- PHP 3, PHP 4: Compile un module Apache. DIR est le dossier d'installation d'Apache (par défaut, /usr/local/etc/httpd).

[4.3.7.4 install.configure.with-apxs](#)

[\[Notes en ligne\]](#)

--with-apxs[=FILE]

- PHP 3, PHP 4: Compile un module partagé Apache. FILE est un chemin optionnel jusqu'aux outils Apache apxs (par défauts, apxs).

[4.3.7.5 install.configure.enable-versioning](#)

[\[Notes en ligne\]](#)

--enable-versioning

- PHP 3: Profite du versioning et du scoping de Solaris 2.x et Linux
PHP 4: Export uniquement les symboles requis. Voir INSTALL pour plus d'informations.

[4.3.7.6 install.configure.with-fhttpd](#)

[\[Notes en ligne\]](#)

--with-fhttpd[=DIR]

- PHP 3, PHP 4: Compile un module fhttpd. DIR est le dossier d'installation de fhttpd (par défaut, /usr/local/src/fhttpd).

[4.3.7.7 install.configure.with-nsapi](#)

[\[Notes en ligne\]](#)

--with-nsapi=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Specifie le chemin jusqu'à Netscape

[4.3.7.8 install.configure.with-phttpd](#)

[\[Notes en ligne\]](#)

--with-phttpd=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4:

[4.3.7.9 install.configure.with-pi3web](#)

[\[Notes en ligne\]](#)

--with-pi3web=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile PHP comme module pour Pi3Web.

[4.3.7.10 install.configure.with-roxen](#)

[\[Notes en ligne\]](#)

--with-roxen=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile PHP comme module pour Pike . DIR est le dossier d'installation de Roxen (par défaut, /usr/local/roxen/server).

[4.3.7.11 install.configure.enable-roxen-zts](#)

[\[Notes en ligne\]](#)

--enable-roxen-zts

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile le module Roxen avec Zend Thread Safety.

[4.3.7.12 install.configure.with-thttpd](#)

[\[Notes en ligne\]](#)

--with-thttpd=SRCDIR

- PHP 3: Option non disponible en PHP 3
PHP 4:

[4.3.7.13 install.configure.with-zeus](#)

[\[Notes en ligne\]](#)

--with-zeus=DIR

- PHP 3: Option non disponible en PHP 3
PHP 4: Compile PHP comme module ISAPI pour Zeus.

[4.3.8 Texte et langue](#)

[\[Notes en ligne\]](#)

[4.3.8.1 install.configure.with-aspell](#)

[\[Notes en ligne\]](#)

--with-aspell[=DIR]

- PHP 3, PHP 4: Inclus le support ASPELL.

[4.3.8.2 install.configure.with-gettext](#)

[\[Notes en ligne\]](#)

--with-gettext[=DIR]

- PHP 3, PHP 4: Inclus le support GNU gettext. DIR est le dossier d'installation de gettext (par défaut, /usr/local).

[4.3.8.3 install.configure.with-pspell](#)

[\[Notes en ligne\]](#)

--with-pspell[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support PSPELL.

[4.3.8.4 install.configure.with-recode](#)

[\[Notes en ligne\]](#)

--with-recode[=DIR]

- PHP 3: Include GNU recode support.
PHP 4: Inclus le support recode. DIR est le dossier d'installation de recode.

[4.3.9 XML](#)

[\[Notes en ligne\]](#)

[4.3.9.1 install.configure.with-dom](#)

[\[Notes en ligne\]](#)

--with-dom[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support DOM (requiert libxml >= 2.0). DIR est le dossier d'installation de libxml (par défaut, ``/usr'`).

[4.3.9.2 install.configure.enable-sablot-errors-descriptive](#)

[\[Notes en ligne\]](#)

--enable-sablot-errors-descriptive

- PHP 3: Option non disponible en PHP 3
PHP 4: Active les erreurs descriptives.

[4.3.9.3 install.configure.with-sablot](#)

[\[Notes en ligne\]](#)

--with-sablot[=DIR]

- PHP 3: Option non disponible en PHP 3
PHP 4: Inclus le support Sablotron

[4.3.9.4 install.configure.enable-wddx](#)

[\[Notes en ligne\]](#)

--enable-wddx

- PHP 3: Option non disponible en PHP 3
PHP 4: Active le support WDDX

[4.3.9.5 install.configure.disable-xml](#)

[\[Notes en ligne\]](#)

--disable-xml

- PHP 3: Option non disponible en PHP 3; Les fonctionnalités XML ne sont pas compilées par défaut.

Utilisez [4.3.9.6 install.configure.with-xml](#).

PHP 4: Désactive le support XML qui utilise la librairie fournie expat.

[4.3.9.6 install.configure.with-xml](#)

[\[Notes en ligne\]](#)

`--with-xml`

- PHP 3: Inclus le support XML
- PHP 4: Option non disponible; XML est compilé par défaut. Utilisez [4.3.9.5 install.configure.disable-xml](#) pour le désactiver.

[4.4 Installation sous Windows 95/98/NT](#)

[\[Notes en ligne\]](#)

Ce guide d'installation vous aidera à installer et configurer PHP sur vos serveurs Windows 9x/NT. Ce guide a été compilé par

Ce guide fourni une aide d'installation pour :

- Personal Web Server (Version la plus récente recommandée)
- Internet Information Server 3 ou 4
- Apache 1.3.x
- Omni HTTPd 2.0b1

[4.4.1 Installation](#)

[\[Notes en ligne\]](#)

Les instructions doivent être faites pour toutes les installations avant d'attaquer les insctructions spécifiques à chaque serveur.

- Extrayez la distribution dans un dossier de votre choix. "C:\PHP\" est une bonne idée.
- Copiez le fichier, 'php.ini-dist' dans le dossier '%WINDOWS%' et renommez le en 'php.ini'.
'%WINDOWS%' est typiquement
 - ◆ c:\windows pour Windows 95/98
 - ◆ c:\winnt ou c:\winnt40 pour les serveurs NT/2000
- Editez votre fichier 'php.ini' :
 - ◆ Vous devez changer votre option 'extension_dir' pour qu'il pointe sur votre dossier d'installation PHP, ou vers l'endroit où vous avez installé vos 'php_*.dll'. ex: c:\php
 - ◆ Si vous utilisez Omni Httpd, sautez l'étape suivante. Modifiez 'doc_root' pour qu'il pointe sur votre racine de serveur web. ex: c:\apache\htdocs ou c:\webroot
 - ◆ Choisissez les modules que vous voulez charger lorsque PHP démarre. Vous pouvez décommenter les lignes 'extension=php_*.dll' pour charger ces modules. Certains modules requièrent que des librairies supplémentaires soient installées sur votre système. La [FAQ](#) PHP a plus d'informations sur ces librairies. Vous pouvez aussi charger dynamiquement ces librairies avec `dl("php_*.dll") ;`
 - ◆ Sous PWS et IIS, vous pouvez modifier le fichier 'browscap.ini' pour qu'il pointe sur :

`c:\windows\system\inetrv\browscap.ini' sous Windows 95/98 et
 `c:\winnt\system32\inetrv\browscap.ini' sous NT Plus de détails sur
 l'utilisation de browscap sont accessibles sur ce [mirroir](#), sélectionnez le bouton "source" pour
 le voir en action.

Les DLLs des extensions PHP sont préfixé avec 'php_', pour éviter les confusions entre les extensions PHP et leur librairies.

4.4.2 Windows 95/98/NT et PWS/IIS 3

[\[Notes en ligne\]](#)

La méthode recommandée pour configurer ces serveurs est d'utiliser le fichier INF inclus dans la distribution (php_iis_reg.inf). Vous pouvez éditer ce fichier, pour vous assurer que les extensions et les dossiers d'installation de PHP sont bien ceux de votre configuration. Ou alors, vous pouvez suivre les instructions suivantes :

ATTENTION: Ces instructions requièrent la manipulation du fichier de registry de Windows. Une erreur peut laisser votre système dans un état instable. Nous vous recommandons vivement de sauvegarder ce fichier en lieu sûr. L'équipe de développement ne pourra pas être reconnue responsable d'un quelconque dommage dans votre registry.

- Lancez Regedit.
- Naviguez jusqu'à : HKEY_LOCAL_MACHINE /System /CurrentControlSet /Services /W3Svc /Parameters /ScriptMap.
- Dans le menu edit, sélectionnez : New->String Value.
- Entrez l'extension que vous voulez utiliser pour les scripts PHP. ex: .php
- Double cliquez sur la chaîne, et entrez le chemin jusqu'à php.exe dans le champ "value data". ex: c:\php\php.exe %s %s. Les '%s %s' sont TRES importants, PHP ne fonctionnera pas sans.
- Répétez ces instructions pour toutes les extensions que vous voulez associer aux scripts PHP.
- Naviguez jusqu'à : HKEY_CLASSES_ROOT
- Dans le menu edit, sélectionnez: New->Key.
- Donnez le nom de votre extension à la clé : ex: .php
- Sélectionnez le nom de la nouvelle clé dans le panneau de droite, et double cliquez dans "default value", puis entrez phpfile.
- Répétez ces instructions pour toutes les extensions que vous avez associé aux scripts PHP.
- Créez une autre New->Key sous HKEY_CLASSES_ROOT et nommez la phpfile.
- Sélectionnez la nouvelle clé phpfile et dans le panneau de droite, double cliquez dans "default value" et entrez PHP Script.
- Faites un clic droit dans phpfile et sélectionnez New->Key, appelez-le Shell.
- Faites un clic droit dans Shell et sélectionnez New->Key, appelez-le open.
- Faites un clic droit dans open et sélectionnez New->Key, appelez-le command.
- Sélectionnez la nouvelle clé command et dans le panneau de droite, faites un double clic dans "default value", puis entrez le chemin jusqu'à php.exe. ex: c:\php\php.exe -q %1. (n'oubliez pas le %1).
- Exit Regedit.

Les utilisateurs de PWS et IIS 3 sont prêts à utiliser leur serveur. Avec IIS 3, vous pouvez utiliser un [outil](#) bien pratique de Steven Genusa pour configurer votre carte des scripts.

4.4.3 Windows NT et IIS 4

[\[Notes en ligne\]](#)

Pour installer PHP sur des serveurs NT avec IIS 4, suivez les instructions suivantes :

- Dans l'Internet Service Manager (MMC), sélectionnez le site web, ou le dossier racine.
- Ouvrez la feuille de propriétés du dossier (avec un clic droit dessus, puis en sélectionnant properties), puis cliquez dans l'onglet Home Directory, Virtual Directory, ou Directory.
- Cliquez dans le bouton Configuration, puis cliquez dans l'onglet App Mappings.
- Cliquez dans la bouton Add, et dans la boîte Executable, tapez: c:\path-to-php-dir\php.exe %s %s. Vous DEVEZ ajouter les %s %s à la fin, car sinon, PHP ne fonctionnera pas sans.
- Dans la boîte Extension, tapez le nom de l'extension de fichier que vous voulez associer à PHP. (Vous devez répéter les étapes 5 et 6 pour toutes les extensions que vous voulez associer à PHP. (.php et .phtml sont les plus répandus).
- Sélectionnez la sécurité appropriée (grâce à l'Internet Service Manager), et si votre serveur NT utilise NTFS, ajoutez les droits d'exécutions pour I_USR_ au dossier qui contient php.exe.

4.4.4 Windows 9x/NT et Apache 1.3.x

[\[Notes en ligne\]](#)

Vous devez éditer srm.conf ou httpd.conf pour configurer Apache, afin qu'il fonctionne avec PHP CGI. Bien qu'il puisse y avoir quelques variations de configurations de PHP sous Apache, elle est suffisamment simple pour être faite par un novice. Reportez vous à la doc Apache pour plus de détails.

- ScriptAlias /php/ "c:/path-to-php-dir/"
- AddType application/x-httpd-php .php
- AddType application/x-httpd-php .phtml
- Action application/x-httpd-php "/php/php.exe"

Pour utiliser la fonction de colorisation de la syntaxe, créez simplement un script PHP, et ajoutez le code suivant : `<?php show_source("original_php_script.php"); ?>`. Substituez original_php_script.php avec le nom du fichier dont vous voulez voir le source (c'est le seul moyen de faire cela). *Note:* Sous Win-Apache tous les antislash (\) dans un chemin de fichier (tel que "c:\directory\file.ext"), doivent être convertis en slash (/).

4.4.5 Omni HTTPd 2.0b1 pour Windows

[\[Notes en ligne\]](#)

La méthode la plus simple pour configurer le serveur est :

- Step 1: Installez Omni server
- Step 2: Faites un clic-droit sur l'icone bleue d'OmniHTTPd, sur le bureau, et sélectionnez Properties
- Step 3: Cliquez sur Web Server Global Settings
- Step 4: Dans l'onglet 'External', entrez: virtual = .php | actual = c:\path-to-php-dir\php.exe
- Step 5: Dans l'onglet Mime, entrez: virtual = wwwserver/stdcgi | actual = .php
- Step 6: Cliquez sur OK

Répez les étapes 2 à 6 pour chaque extension que vous voulez associer à PHP.

4.4.6 Installshield

[\[Notes en ligne\]](#)

L'installateur PHP pour Windows PHP, disponibles depuis les pages de téléchargement à <http://www.php.net/> installe la version CGI de PHP et, pour IIS, PWS, et Xitami, configure le serveur web en même temps.

Installez votre serveur **HTTP** sur votre système, puis assurez vous qu'il fonctionne.

Lancez l'installateur (.exe), et suivez les instructions fournies par le wizard. Deux types d'installations sont supportés : standard, qui effectue une configuration standard, et avancé, qui demande la configuration au fur et à mesure.

Le wizard d'installation rassemble suffisamment d'informations pour configurer le fichier `php.ini` et configurer le serveur web pour qu'il utilise PHP. Pour IIS et PWS sous NT Workstation, il affiche une liste de tous les noeuds du serveur, avec leur configuration. Vous pouvez alors choisir quels noeuds bénéficieront de la configuration PHP.

Une fois l'installation complète, l'installateur indiquera qu'il faut redémarrer votre système, et le fera pour voir. Ou bien, vous pourrez immédiatement utiliser PHP.

4.4.7 Modules PHP

[\[Notes en ligne\]](#)

php_calendar.dll	Fonctions de conversions calendaires
php_crypt.dll	Fonctions de cryptage
php_dbase.dll	Fonctions DBase
php_dbm.dll	Librairie d'émulation GDBM via Berkely DB2
php_filepro.dll	Lecture des bases filepro
php_gd.dll	Bibliothèque GD (pour les manipulations d'images)
php_hyperwave.dll	Fonctions HyperWave
php_imap4r2.dll	Fonctions IMAP 4
php_ldap.dll	Fonctions LDAP
php_msql1.dll	Fonctions mSQL 1
php_msql2.dll	Fonctions mSQL 2
php_mssql.dll	Fonctions MSSQL (requiert MSSQL DB-Libraries)
php3_mysql.dll (compilé dans PHP 4)	Fonctions MySQL
php_nsmail.dll	Fonctions Netscape mail
php_oci73.dll	Fonctions Oracle
php_snmp.dll	Fonctions SNMP get et walk (NT uniquement!)
php_zlib.dll	Fonctions ZLib

4.5 Problèmes?

[\[Notes en ligne\]](#)

4.5.1 Lisez la FAQ

[\[Notes en ligne\]](#)

Certains problèmes sont récurrents : Les plus commun sont listés dans la FAQ PHP, disponible à <http://www.php.net/FAQ.php>

4.5.2 Rapport de bug

[\[Notes en ligne\]](#)

Si vous pensez avoir trouvé un bug dans PHP, n'oubliez pas de le signaler. L'équipe de développement PHP ne le connaît peut être pas, et sans le signaler, vos chances de voir le bug corrigés sont nulles. Vous pouvez rapporter des bugs grâce au système de suivi, accessible à <http://www.php.net/bugs.php>.

4.5.3 Autres problèmes

[\[Notes en ligne\]](#)

Si vous êtes complètement bloqués, quelqu'un sur la liste de diffusion PHP pourra probablement vous aider. Essayez de consulter les archives, au cas où quelqu'un aurait déjà rencontré votre problème. Les archives sont toujours accessibles à : <http://www.php.net/>. Pour souscrire à la liste de diffusion, envoyez un mail vide à L'adresse de la mailing liste : php-general@lists.php.net.

Si vous voulez obtenir de l'aide sur la liste de diffusion PHP, essayez d'être concis et clair, et pensez à donner tous les détails sur votre environnement (OS, version de PHP, serveur web, CGI ou module, [7.1.3.1 ini.safe-mode...](#)), et n'hésitez pas à envoyer suffisamment de code pour que nous puissions reproduire l'erreur.

5 Introduction

[\[Notes en ligne\]](#)

5.1 Qu'est ce que PHP?

[\[Notes en ligne\]](#)

PHP (officiellement "PHP: Hypertext Preprocessor") est un langage de script HTML, qui fonctionne coté serveur.

Réponse simple et claire, mais qu'est ce que cela veut dire? Un exemple :

Exemple d'introduction

```
<html>
  <head>
    <title>Exemple</title>
  </head>
  <body>
    <?php
echo "Bonjour, je suis un script PHP!";
    ?>
  </body>
</html>
```

Il est à noter la différence avec les autres scripts CGI écrit dans d'autres langages tels que le Perl ou le C : Au lieu d'écrire un programme avec de nombreuses lignes de commandes afin d'afficher une page HTML, vous écrivez une page HTML avec du code inclus à l'intérieur afin de réaliser une action précise (dans ce cas là, afficher du texte). Le code PHP est inclus entre [9.1.1 Le passage du HTML au PHP](#) qui permettent au navigateur de passer en "mode PHP".

Ce qui distingue le PHP des langages de script comme le Javascript est que le code est exécuté sur le serveur. Si vous avez un script similaire sur votre serveur, le client ne reçoit que le résultat du script, sans aucun moyen d'avoir accès au code qui a produit ce résultat. Vous pouvez configurer votre serveur web afin qu'il analyse tous vos fichiers HTML comme des fichiers PHP. Ainsi, il n'y a aucun moyen de distinguer les pages qui sont produites dynamiquement des pages statiques.

5.2 Que peut faire PHP?

[\[Notes en ligne\]](#)

Le langage PHP possède les même fonctionnalités que les autres langages permettant d'écrire des scripts CGI, comme collecter des données, générer dynamiquement des pages web ou bien envoyer et recevoir des cookies.

La plus grande qualité et le plus important avantage du langage PHP est le support d'un grand nombre de bases de données. Réaliser une page web dynamique interfacant une base de données est extrêmement simple. Les bases de données suivantes sont supportées par PHP:

- Adabas D
- dBase

- Empress
- FilePro (lecture seule)
- Hyperwave
- IBM DB2
- Informix
- Ingres
- InterBase
- FrontBase
- mSQL
- Direct MS–SQL
- MySQL
- ODBC
- Oracle (OCI7 et OCI8)
- Ovrinos
- PostgreSQL
- Sesam
- Solid
- Sybase
- Velocis
- Unix dbm

Le langage PHP inclut le support des services utilisant les protocoles tels que *IMAP*, *SNMP*, *NNTP*, *POP3* ou encore *HTTP*. Vous pouvez également ouvrir des connexions et interagir en utilisant d'autres protocoles.

5.3 La genèse du PHP

[\[Notes en ligne\]](#)

Le langage PHP a été conçu durant l'automne 1994 par Rasmus Lerdorf. Les premières versions (qui restèrent privées) étaient utilisées afin de savoir qui venait consulter son CV en ligne. La première version publique fut disponible au début de l'année 1995. Elle fut connue sous le nom de "Personal Sommaire Page Tools". Elle était composée d'un analyseur extrêmement simple qui ne reconnaissait que quelques macros spéciales et d'un petit nombre d'utilitaires couramment utilisés dans les pages web. Un livre d'or, un compteur, etc...

L'analyseur fut réécrit durant l'été 1995 et fut appelé PHP/FI Version 2. FI étaient les initiales d'un autre package que Rasmus avait écrit qui interprétait les formulaires HTML. C'est alors qu'il combina le "Personal Sommaire Page tools" avec le "Form Interpreter" et il y ajouta le support de mSQL: c'est comme cela que naquit PHP/FI. PHP/FI grandit de manière spectaculaire et de nombreuses personnes commencèrent à contribuer à son amélioration.

Il est relativement peu aisé de donner des statistiques, mais on estime que PHP/FI est utilisé sur 15 000 sites web dans le monde entier, fin 1996. Ce chiffre atteint 50 000 durant l'été 1997. L'été 1997 voit aussi un profond changement dans le développement du PHP: d'un projet personnel (celui de Rasmus), on passe alors à un projet d'équipe. L'analyseur fut de nouveau réécrit par Zeev Suraski et Andi Gutmans et ce nouvel analyseur forma la base de la version 3 du PHP. Une grande partie du code de PHP/FI fut complètement réécrit alors que l'autre partie fut portée pour donner le PHP Version 3. La dernière version de PHP (PHP 4) utilise le moteur d'analyse [Zend](#) pour atteindre de nouveaux niveaux de performance, et supporter un nombre encore plus grand de bibliothèques et extensions. Il tourne de manière native sur tous les serveurs web les plus répandus.

Aujourd'hui (Janvier 2001) PHP 3 ou PHP 4 sont distribués avec de nombreux produits commerciaux comme "C2's StrongHold web server" et "RedHat Linux" et il est admis (d'après les chiffres de [NetCraft](#), et leurs

[statistiques Netcraft Web Server Survey](#)) que le PHP est utilisé sur 5 100 000 sites web dans le monde entier. Pour comparaison, ce chiffre est légèrement supérieur au nombre de serveurs tournant sous Microsoft Information server (IIS) : 5.03 millions. Enfin, à l'heure où ce document est rédigé, la nouvelle génération du PHP est en cours de création. Elle utilisera les qualités de [Zend](#) pour améliorer les performances et améliorera le support des serveurs web autres que Apache.

6 Sécurité

[\[Notes en ligne\]](#)

PHP est un langage puissant et l'interpréteur, qu'il soit inclus dans le serveur web ou bien compilé en version CGI, est capable d'accéder aux fichiers, d'exécuter des commandes et d'ouvrir des connexions réseaux. Toutes ces propriétés rendent fragile la sécurité d'un serveur web. Le langage PHP a été pensé afin d'être un langage beaucoup plus sécurisé pour écrire des **CGI** que le Perl ou le langage C. De plus une sélection rigoureuse des options de compilation et d'exécution vous permettront d'obtenir un équilibre parfait entre liberté et sécurité. Etant donné qu'il y a de nombreux moyens d'utiliser le langage PHP, il y a de nombreuses directives de configuration afin d'en contrôler le comportement. Un grand nombre d'options permettent d'utiliser le PHP dans de nombreuses situations, mais cela signifie aussi qu'il y a certaines combinaisons d'options de compilation et d'exécution qui fragilise la sécurité du serveur. Ce chapitre explique comme les différentes options de configurations peuvent être combinées, tout en conservant une sécurité maximum. La flexibilité de configuration de PHP est épaulée par la flexibilité du code. PHP peut être compilé pour constituer une application serveur complète, avec toutes les fonctionnalités d'un shell, ou il peut encore être utilisé comme simple SSI (server side include) avec peu de risque, dans un environnement à sécurité renforcée. Comment créer cet environnement et le sécuriser est largement à la charge du développeur PHP. Ce chapitre commence par expliquer les différentes options de configuration et les situations où elles peuvent être utilisées en toute sécurité. Puis, viennent les considérations de niveaux de sécurité, et les conseils généraux.

6.1 Binaires CGI

[\[Notes en ligne\]](#)

6.1.1 Faiblesses connues

[\[Notes en ligne\]](#)

Utiliser le PHP comme un **CGI** exécutable vient la plupart du temps du fait que l'on ne veut pas l'utiliser comme un module du serveur web, (comme Apache), ou bien que l'on souhaite l'utiliser en combinaison d'un **CGI** complémentaire, afin de créer un environnement de script sécurisé (en utilisant des techniques de chroot ou setuid). Une telle décision signifie habituellement que vous installez votre exécutable dans le répertoire cgi-bin de votre serveur web. [CERT CA-96.11](#) recommande effectivement de placer l'interpréteur à l'intérieur du répertoire cgi-bin. Même si le binaire PHP peut être utilisé comme interpréteur indépendant, PHP a été pensé afin de rendre impossible les attaques que ce type d'installation induit.

- Accès au système de fichier: ``http://ma.machine/cgi-bin/php?/etc/passwd``
Lorsque la requête est passée dans une url, après le point d'interrogation (?), elle est envoyée à l'interpréteur comme une ligne de commande par l'interface CGI. Habituellement, l'interpréteur ouvre le fichier spécifié et l'exécute.
Lorsqu'il est invoqué comme exécutable CGI, le PHP refuse d'interpréter les arguments de la ligne de commande.
- Accès d'un document web sur le serveur :
``http://my.host/cgi-bin/php/secret/doc.html``
Le "path information" dans l'url, situé juste après le nom du binaire PHP, ``/secret/doc.html`` est utilisé par convention pour spécifier le nom du fichier qui doit être ouvert et interprété par le programme **CGI**. Habituellement, des directives de configuration du serveur web (pour le serveur Apache: Action) sont utilisées pour rediriger les requêtes pour obtenir un document ``http://my.host/secret/script.php`` par l'interpréteur PHP. Dans une telle

configuration, le serveur web vérifie d'abord si il a accès au répertoire ``/secret'`, et après cette vérification redirige la requête vers ``http://my.host/cgi-bin/php/secret/script.php'`. Malheureusement, si la requête est faite directement sous cette forme, aucune vérification d'accès n'est faite par le serveur web pour le fichier ``/secret/script.php'`, mais uniquement pour le fichier ``/cgi-bin/php'`. De cette manière, n'importe quel utilisateur qui peut accéder au fichier ``/cgi-bin/php'` peut aussi accéder au document protégés sur le serveur web.

Avec le PHP, l'option de compilation [4.3.4.19 install.configure.enable-force-cgi-redirect](#) et les options d'exécution [7.1.1.7 ini.doc-root](#) et [7.1.1.29 ini.user-dir](#) peuvent être utilisées pour prévenir ce genre d'attaques, si des restrictions d'accès sont appliquées sur les documents du serveur. Voir ci-dessous pour des explications plus complètes sur les différentes combinaisons.

6.1.2 Cas 1: Tous les fichiers sont publics

[\[Notes en ligne\]](#)

Si votre serveur n'a aucun document dont l'accès est restreint par un mot de passe ou un système de vérification de l'adresse IP, vous n'avez aucun besoin de ce type de configuration. Si votre serveur web ne permet pas les redirections, ou si votre serveur web n'a aucun besoin de communiquer avec le binaire PHP de manière sécurisée, vous pouvez utiliser l'option de compilation [4.3.4.19 install.configure.enable-force-cgi-redirect](#). Vous devez quand même vérifier qu'aucun script ne fait appel au PHP, de manière directe, ``http://my.host/cgi-bin/php/dir/script.php'` ou bien de manière indirecte, par redirection, ``http://my.host/dir/script.php'`.

Les redirections peuvent être configurées dans les fichiers de configuration d'Apache en utilisant les directives "AddHandler" et "Action" (voir ci-dessous).

6.1.3 Cas 2: Utilisation de la directive de compilation `--enable-force-cgi-redirect`

[\[Notes en ligne\]](#)

Cette option de compilation prévient quiconque d'appeler directement un script avec l'url ``http://my.host/cgi-bin/php/secret/dir/script.php'`. Dans ce cas là, PHP parsera le fichier uniquement si il y a eu redirection.

Habituellement, le serveur web Apache réalise une redirection grâce aux directives suivantes :

```
Action.php-script /cgi-bin/php
AddHandler.php-script .php
```

Cette option a uniquement été testée avec Apache et compte sur Apache pour affecter la variable d'environnement non-standart **REDIRECT_STATUS** pour les requêtes redirigées. Dans le cas où votre serveur web ne supporte pas le renseignement du PHP, pour savoir si la requête a été redirigée ou non, vous ne pouvez pas utiliser cette option de compilation. Vous devez alors utiliser une des autres manières pour utiliser la version binaire CGI du PHP, comme exposé ci-dessous.

6.1.4 Cas 3: Utilisation du "doc_root" ou du "user_dir"

[\[Notes en ligne\]](#)

Ajouter un contenu interactif dans votre serveur web, comme des scripts ou des exécutables, est souvent considéré comme une pratique non-sécurisée. Si, par erreur, le script n'est pas exécuté mais affiché comme une page HTML classique, il peut en résulter un vol de propriété intellectuelle ou des problèmes de sécurité à

propos des mots de passe notamment. Donc, la plupart des administrateurs préfèrent mettre en place un répertoire spécial pour les scripts qui est uniquement accessible par le biais du binaire CGI du PHP, et donc, tous les fichiers de ce répertoire seront interprétés et non affichés tels quel.

Aussi, si vous ne pouvez pas utiliser la méthode présentée ci-dessus, il est nécessaire de mettre en place un répertoire "doc_root" différent de votre répertoire "document root" de votre serveur web.

Vous pouvez utiliser la directive [7.1.1.7 ini.doc-root](#) dans le [7.1 Le fichier de configuration](#), ou vous pouvez affecter la variable d'environnement **PHP_DOCUMENT_ROOT**. Si cette variable d'environnement est affectée, le binaire **CGI** du PHP construira toujours le nom de fichier à ouvrir avec **doc_root** et le "path information" de la requête, et donc vous serez sûr qu'aucun script n'est exécuté en dehors du répertoire prédéfini. (à l'exception du répertoire désigné par la directive **user_dir** Voir ci-dessous).

Une autre option possible ici est la directive [7.1.1.29 ini.user-dir](#). Lorsque la directive n'est pas activée, seulement les fichiers contenus dans le répertoire **doc_root** peuvent être ouverts. Ouvrir un fichier possédant l'url ``http://my.host/~user/doc.php`` ne correspond pas à l'ouverture d'un fichier sous le répertoire racine de l'utilisateur mais à l'ouverture du fichier ``~user/doc.php`` sous le répertoire "doc_root" (oui, un répertoire commence par un tilde [~]).

Si la directive "user_dir" est activée à la valeur ``public_php`` par exemple, une requête du type ``http://my.host/~user/doc.php`` ouvrira un fichier appelé ``doc.php`` sous le répertoire appelé ``public_php`` sous le répertoire racine de l'utilisateur. Si le répertoire racine des utilisateurs est ``/home/user``, le fichier exécuté sera ``/home/user/public_php/doc.php``.

user_dir et **doc_root** sont deux directives totalement indépendantes et donc vous pouvez contrôler l'accès au répertoire "document root" séparément des répertoires "user directory".

[6.1.5 Cas 4: L'exécutable PHP à l'extérieur de l'arborescence du serveur](#)

[\[Notes en ligne\]](#)

Une solution extrêmement sécurisée consiste à mettre l'exécutable PHP à l'extérieur de l'arborescence du serveur web. Dans le répertoire ``/usr/local/bin``, par exemple. Le problème de cette méthode est que vous aurez à rajouter la ligne suivante :

```
#!/usr/local/bin/php
```

dans tous les fichiers contenant des tags PHP. Vous devrez aussi rendre le binaire PHP exécutable. Dans ce cas-là, traitez le fichier exactement comme si vous aviez un autre script écrit en Perl ou en sh ou en un autre langage de script qui utilise `#!` comme mécanisme pour lancer l'interpréteur lui-même.

Pour que l'exécutable PHP prenne en compte les variables d'environnement **PATH_INFO** et **PATH_TRANSLATED** correctement avec cette configuration, vous devez utiliser l'option de compilation [4.3.4.14 install.configure.enable-discard-path](#).

[6.2 Module Apache](#)

[\[Notes en ligne\]](#)

Lorsque le PHP est compilé en tant que module Apache, ce module hérite des permissions accordées à l'utilisateur faisant tourner Apache (par défaut, l'utilisateur "nobody"). Par exemple, si vous utilisez PHP pour accéder à une base de données à moins que la base n'ai un système de droits d'accès interne, vous devrez rendre la base accessible à l'utilisateur "nobody". Cela signifie qu'un script malintentionné peut accéder à la base, la modifier sans authentification. Il est aussi possible qu'un robot accède à la page d'administration, et détruise toutes les pages. Vous devez ainsi protéger vos bases de données avec les autorisations Apache, ou définir votre propre modèle d'accès avec LDAP, .htaccess, etc... et include ce code dans tous vos scripts PHP. Souvent, lorsqu'on a établi les droits pour que l'utilisateur PHP (ici, l'utilisateur Apache) pour minimiser les

risques, on s'aperçoit que PHP ne peut plus écrire des virus dans les fichiers des utilisateurs. Ou encore, de modifier une base de données privée. Il est aussi incapable de modifier des fichiers qu'il devrait pouvoir modifier, ou effectuer certaines transactions.

Une erreur fréquente de sécurité est de donner à l'utilisateur Apache les droits de superadministrateur. Donner de telles permissions à l'utilisateur Apache est extrêmement dangereux, et peut compromettre tout le système, tel que l'utilisation des sudo ou du chroot. Une telle utilisation est à exclure pour les professionnels de la sécurité.

6.3 Sécurité des fichiers

[\[Notes en ligne\]](#)

PHP est soumis aux règles de sécurité intrinsèques de la plus part des systèmes serveurs : il respecte notamment les droits des fichiers, et des dossiers. Une attention particulière doit être portée aux fichiers qui sont accessibles à tout le monde, afin de s'assurer qu'ils ne divulguent pas d'informations critiques.

Puisque PHP a été fait pour permettre aux utilisateurs d'accéder aux fichiers, il est possible de créer un script qui vous permet de lire des fichiers tels que /etc/password, de modifier les connexions ethernet, lancer des impressions de documents, etc... Cela implique notamment que vous devez vous assurer que les fichiers accédés par les scripts sont bien ceux qu'il faut.

Considérez le script suivant, où l'utilisateur indique qu'il souhaite effacer un fichier dans son dossier racine. Nous supposons que PHP est utilisé comme interface web pour gérer les fichiers, et que l'utilisateur Apache est autorisé à effacer les fichiers dans le dossier racine des utilisateurs.

Une erreur de vérification de variable conduit à

```
<?php
// efface un fichier dans un dossier racine
$username = $user_submitted_name;
$homedir = "/home/$username";
$file_to_delete = "$userfile";
unlink ($homedir/$userfile);
echo "$file_to_delete a été effacé!";
?>
```

Etant donné que le nom de l'utilisateur est à fournir, ils peuvent envoyer un nom d'utilisateur autre que le leur, et effacer des documents dans les comptes des autres utilisateurs. Dans ce cas, vous souhaitez utiliser une autre forme d'authentification. Considérez ce qui pourrait se passer si les utilisateurs passent "../etc/" et "passwd" comme arguments!. Le code serait exécuté tel que :

Une attaque du système de fichier!

```
<?php
// efface un fichier n'importe où sur le disque dur,
// où l'utilisateur PHP a accès. Si PHP a un accès root :
$username = "../etc/";
$homedir = "/home/../etc/";
$file_to_delete = "passwd";
unlink ("/home/../etc/passwd");
echo "/home/../etc/passwd" has been deleted!";
?>
```

Il y a deux mesures primordiales à prendre pour éviter ces manoeuvres :

- Limiter les permissions du l'utilisateur web PHP.
- Vérifier toutes les variables liées aux chemins et aux fichiers qui sont fournies.

Voici le script renforcé :

Une vérification renforcée

```
<?php
// Efface un fichier sur le disque où l'utilisateur a le droit
$username = get_env("REMOTE_USER");
// utilise un mécanisme d'authentification
$homedir = "/home/$username";
$file_to_delete = basename("$userfile");
// supprime le chemin excédentaire
unlink ($homedir/$file_to_delete);
$fvp = fopen("/home/logging/filedelete.log", "a"); //note l'effacement
$logstring = "$HTTP_REMOTE_USER $homedir $file_to_delete";
fputs ($fvp, $logstring);
fclose($fvp);
echo "$file_to_delete a été effacé!";
?>
```

Vous pouvez vous protéger avec une vérification telle que :

Vérification de noms de fichier sécurisée

```
<?php
$username = $HTTP_REMOTE_USER;
$homedir = "/home/$username";
if (!ereg('^[^./][^/]*$', $userfile))
die('Erreur de nom de fichier'); //meurt, ne pas traiter!
//etc...
?>
```

Suivant votre système d'exploitation, vous devrez protéger un grand nombre de fichiers, notamment les entrées de périphériques, (/dev/ ou COM1), les fichiers de configuration (fichiers /etc/ et .ini), les lieux de stockage d'informations (/home/, My Documents), etc. Pour cette raison, il est généralement plus sûr d'établir une politique qui interdit TOUT sauf ce que vous autorisez.

6.4 Rapport d'erreur

[\[Notes en ligne\]](#)

Une tactique d'attaque standard consiste à faire faire des erreurs au système, et lire les variables d'environnement et de contexte qui sont retournées. Cela permet au hacker de lire de nombreuses informations sur le serveur, et détecter des faiblesses du serveur.

Les erreurs PHP qui sont normalement retournées peuvent être très pratiques pour un développeur qui essaie de déboguer un scripts, car elles donnent de précieux renseignements tels que quelle fonction a échoué, quel fichier n'a pas été trouvé, quel script PHP a buggé, et quelle ligne est en faute. Toutes ces informations sont exploitables. Il est commun aux développeurs PHP d'utiliser les fonctions [show_source\(\)](#), [highlight_string\(\)](#), ou [highlight_file\(\)](#) comme outils de débogage, mais sur un site en production, cela expose des variables cachées, des syntaxes non vérifiées ou d'autres informations dangereuses.

Par exemple, le style d'erreur indique sur quel système PHP fonctionne. Si un pirate affiche une page *html*, et essaye de la tester (pour rechercher des faiblesses du système), il peut déterminer sur quel système PHP a été compilé.

Une erreur de fonction indique si un système supporte une base de données spécifique, ou bien indique comment la page a été générée. Cela peut orienter l'intrus vers les ports de cette base de données ou bien vers une attaque liée à cette application. En envoyant des données erronées, par exemple, un pirate peut déterminer l'ordre d'authentification dans un script (à partir des lignes d'erreurs), et essayer de les exploiter ailleurs, dans le script.

Une erreur de fichier, ou une erreur générale PHP peut indiquer quelles sont les permissions du serveur web, ainsi que la structure et l'organisation des fichiers. Les gestionnaires d'erreurs utilisateurs peuvent aussi aggraver ce problème, en permettant l'exploitation facile d'informations préalablement cachées.

Il y a trois solutions majeures à ces problèmes : la première est de scruter toutes les fonctions, et essayer de traiter toutes les erreurs. La deuxième est d'inactiver le rapport d'erreur, dès que le script est en production. La troisième est d'utiliser les fonctions de gestions des erreurs. Suivant votre politique de sécurité, vous pouvez utiliser un panachage savant des trois méthodes.

6.5 User Submitted Data

[\[Notes en ligne\]](#)

Les plus grandes faiblesses de nombreux programmes PHP ne viennent pas du langage lui-même, mais de son utilisation en omettant les caractéristiques de sécurité. Pour cette raison, vous devez toujours prendre le temps de prendre en compte les implications d'une fonction, et de cerner toutes les applications d'une utilisation exotiques des paramètres.

Utilisation dangereuse de variables

```
<?php
// efface un fichier à la racine d'un utilisateur... ou peut être
// de quelqu'un d'autre?
unlink ($evil_var);
// Note l'accès de ce fichier ... ou pas?
fputs ($fp, $evil_var);
// Exécute une commande triviale... ou pas?
system ($evil_var);
exec ($evil_var);
?>
```

Il est vivement recommandé d'examiner minutieusement votre code pour vous assurer qu'il n'y a pas de variables envoyées par le client web, et qui ne sont pas suffisamment vérifiées avant utilisation.

- Est ce que ce script n'affectera que les fichiers prévus?
- Est il possible que des valeurs incohérentes soient exploitées ici?
- Est ce que ce script peut être utilisé dans un but différents?
- Est ce que ce script peut être utilisé malicieusement, en conjonction avec d'autres?
- Est ce que toutes les actions seront notées?

En répondant de manière adéquate à ces questions lors de l'écriture de vos scripts (plutôt qu'après), vous éviterez une réécriture inopportune pour raison de sécurité. En commençant vos projets avec ces recommandations en têtes, vous garantirez pas la sécurité de votre système, mais vous contribuerez à l'améliorer.

Vous pouvez aussi envisager de supprimer l'acquisition automatique des variables d'environnement, les guillemets magiques (magic_quotes), ou encore toute option qui pourrait vous conduire à mésévaluer la validité, la source ou la valeur d'une variable. En travaillant avec `error_reporting(E_ALL)`, vous pouvez être averti que certaines variables sont utilisées avant d'être exploitées, ou vérifiées (et donc, vous pourrez traiter des valeurs exotiques à la source).

6.6 Considérations générales

[\[Notes en ligne\]](#)

Un système complètement sécurisé est virtuellement impossible. De ce fait, une approche qui équilibre les

risques et l'ergonomie est souvent retenue. Si toutes les variables envoyées par l'utilisateur nécessitaient deux formes d'identification biométriques (comme par exemple un scanner de la rétine, et une empreinte digitale), vous pourriez avoir un niveau de sécurité élevé. Cela prendrait aussi une demi-heure pour remplir les conditions de sécurité, ce qui pousserait les utilisateurs à éviter d'utiliser votre système.

La meilleure sécurité est celle qui protège votre système, sans empêcher les utilisateurs de faire leur travail.

En général, les attaques exploitent des trous de sécurité entre diverses couches d'identification : cet empilement devient trop complexe, et finalement, peu fiable.

Une phrase qui vaut la peine d'être retenue : un système est aussi fiable que son maillon le plus faible. Si toutes les transactions sont exhaustivement notées (heure, lieu, type, etc...) mais que l'utilisateur n'est authentifié que par un cookie, la validité de votre système de surveillance est intimement liée à la validité du cookie (et donc, sévèrement réduite).

Lors de vos tests, gardez à l'esprit que vous ne pourrez pas tester toutes les configurations, mais probablement les plus simples. Les données en entrées auxquelles vous pouvez vous attendre ne seront rien comparées aux données incohérentes qu'un employé négligent, un hacker disposant d'autant de temps qu'il veut, ou du chat de la maison marchant sur le clavier. Il est donc bon de regarder le code logiquement, de voir d'où des données incohérentes peuvent être introduites, modifiées, réduites ou amplifiées.

L'Internet est rempli de personnes qui tentent de se faire une renommée en piratant vos programmes, bloquant votre site, envoyant des contenus inappropriés, qui rendent vos journées si "spéciales". Peu importe que vous ayez un grand portail ou un petit web, vous pouvez être la cible pour tout quidam avec une connexion. Vous êtes une cible potentielle dès que vous êtes connecté vous-même. Certains programmes de piratage ne font pas dans la demi-mesure, et testent systématiquement des millions d'IP, à la recherche de victimes : ne soyez pas la prochaine.

6.7 Etre à jour

[\[Notes en ligne\]](#)

PHP, comme de nombreux systèmes de grandes tailles, est constamment testé et amélioré. Chaque nouvelle version rassemble des modifications majeures et mineures, aussi bien pour renforcer la sécurité, réparer les problèmes de conceptions de configuration, et d'autres points qui peuvent affecter la sécurité globale et la stabilité de votre système.

Comme les autres langages de scripts systèmes, la meilleure approche est de mettre à jour souvent PHP, et de rester à l'écoute des dernières versions et des modifications qu'elles apportent.

7 Configuration

[\[Notes en ligne\]](#)

7.1 Le fichier de configuration

[\[Notes en ligne\]](#)

Le fichier de configuration (appelé ``php3.ini`` dans la version 3.0 du PHP, et simplement ``php.ini`` dans la version 4.0) est lu par le PHP au démarrage. Si vous avez compilé PHP en module, le fichier n'est lu qu'une seule fois, au lancement du démon **HTTP**. Pour la version **CGI** le fichier est lu à chaque invocation.

Lorsque vous utilisez le module Apache vous pouvez aussi changer les paramètres de configurations en utilisant les directives dans les fichiers de configuration d'Apache et dans les fichiers ``.htaccess``. Dans la version 3.0, à chaque directive de configuration présente dans le fichier de configuration d'Apache correspond une directive de configuration dans le fichier ``php3.ini`` à l'exception des directives préfixées par `"php3_"`.

Dans la version 4.0, il n'y a seulement que quelques directives dans le fichier de configuration d'Apache qui vous permettent de modifier la configuration de PHP.

`php_value` *name value*

- Cette directive affecte une valeur à la variable spécifiée.
`php_flag` *name on/off*
- Cette directive est utilisée pour activer ou désactiver une option.
`php_admin_value` *name value*
- Cette directive affecte une valeur à la variable spécifiée. La directive "Admin" ne peut être utilisée que dans le fichier de configuration d'Apache, et non dans un fichier ``.htaccess``.
`php_admin_flag` *name on/off*
- Cette directive est utilisée pour activer ou désactiver l'option précédente.

Vous pouvez voir l'état de votre configuration en utilisant la fonction [phpinfo\(\)](#). Vous pouvez aussi accéder aux valeurs de votre configuration de manière individuelle en utilisant la fonction [get_cfg_var\(\)](#).

7.1.1 Directives de configuration générale

[\[Notes en ligne\]](#)

[7.1.1.1 ini.allow-url-fopen](#)

[\[Notes en ligne\]](#)

`allow_url_fopen` boolean

- Cette option autorise les accès au réseau des fonctions [fopen\(\)](#). Par défaut, l'accès est autorisé aux procédures d' [8.8 Utilisation des fichiers à distance](#), avec les protocoles **FTP**, **NNTP**, et certaines extensions telles que zlib.
Note : Cette option a été introduite immédiatement après la version 4.0.3. Pour les versions jusqu'à la 4.0.3 inclus, vous pouvez désactiver cette fonctionnalité au moment de la compilation en utilisant la configuration [4.3.5.4 install.configure.disable-url-fopen-wrapper](#).

[7.1.1.2 ini.asp-tags](#)

[\[Notes en ligne\]](#)

asp_tags booléen

- Active l'utilisation des balises de type ASP <% %>, en plus des traditionnelles balises <?php ?>. Cela inclut l'utilisation du raccourci <%= \$value %>. Pour plus d'informations, reportez vous à [9.1.1 Le passage du HTML au PHP](#).

Note : *Le support des balises ASP a été ajouté dans la version 3.0.4.*

[7.1.1.3 ini.auto-append-file](#)

[\[Notes en ligne\]](#)

auto_append_file chaîne de caractères

- Spécifie le nom d'un fichier qui sera automatiquement ajouté après le fichier principal. Le fichier est inclus comme si il avait été appelé avec la fonction [include\(\)](#), donc [7.1.1.14 ini.include-path](#) est utilisé.

Le mot réservé NONE désactive l'auto-appending.

Note : *Si le script s'arrête par la fonction [exit\(\)](#), auto-append ne fonctionnera pas.*

[7.1.1.4 ini.auto-prepend-file](#)

[\[Notes en ligne\]](#)

auto_prepend_file chaîne de caractères

- Spécifie le nom d'un fichier qui sera automatiquement ajouté avant le fichier principal. Le fichier est inclus comme si il avait été appelé avec la fonction [include\(\)](#), donc [7.1.1.14 ini.include-path](#) est utilisé.

Le mot réservé NONE désactive l'auto-appending.

[7.1.1.5 ini.cgi-ext](#)

[\[Notes en ligne\]](#)

cgi_ext chaîne de caractères

- (NDT : aucune documentation n'est fournie).

[7.1.1.6 ini.display-errors](#)

[\[Notes en ligne\]](#)

display_errors booléen

- Cette directive détermine si les erreurs doivent être affichées à l'écran au format HTML ou non.

[7.1.1.7 ini.doc-root](#)

[\[Notes en ligne\]](#)

doc_root string

- Le dossier racine de PHP. Utilisée uniquement si elle est définie. Si PHP est configuré en [7.1.3.1](#)

[ini.safe-mode](#), aucun fichier en dehors de ce dossier ne sera accessible.

[7.1.1.8 ini.engine](#)

[\[Notes en ligne\]](#)

engine boolean

- Cette directive ne sert vraiment que si PHP est un module Apache. Elle sert aux sites qui veulent activer ou désactiver l'analyse des fichiers par PHP, dossier par dossier. En mettant `php3_engine off` au bon endroit, dans le fichier ``httpd.conf'`, PHP peut être activé ou désactivé.

[7.1.1.9 ini.error-log](#)

[\[Notes en ligne\]](#)

error_log string

- Nom du fichier où les erreurs seront enregistrées. Si la valeur spéciale `syslog` est utilisée, les erreurs sont envoyées au système standard d'historique. Sous UNIX, c'est `syslog(3)` et sous Windows NT c'est l'historique d'événements. L'historique système n'est pas supporté sous Windows 95.

[7.1.1.10 ini.error-reporting](#)

[\[Notes en ligne\]](#)

error_reporting integer

- Fixe le niveau d'erreur. Ce paramètre est un entier, représentant un champs de bits. Ajoutez les valeurs suivantes pour choisir le niveau que vous désirez :

valeur du bit	niveau choisi
1	erreurs normales
2	alertes normales
4	erreurs d'analyseur (parser errors)
8	alertes non critiques

Par défaut, la valeur est de 7 (erreurs normales, alertes normales et erreurs d'analyseur sont affichées).

[7.1.1.11 ini.open-basedir](#)

[\[Notes en ligne\]](#)

open_basedir string

- Limite l'espace où PHP peut ouvrir des fichiers.
Lorsqu'un script essaie d'ouvrir un fichier avec les fonctions `fopen` ou `gzopen` (par exemple), la localisation du fichier est vérifiée. Si ce fichier est hors du dossier cité dans cette directive, PHP refusera de l'ouvrir. Tous les liens symboliques sont résolus, et subissent aussi la restriction. La valeur spéciale `.` indique que le dossier courant du script est utilisé comme `open_basedir`. Sous Windows, séparez les noms de dossiers par un point virgule (;). Sur les autres systèmes, séparez les noms de dossiers par des deux points (:). Lorsque PHP est un module Apache, la valeur de la directive `open_basedir` des dossiers parents sont automatiquement hérités par les fils.
Note : *Le support pour les dossiers multiples a été ajouté dans 3.0.7.*

La valeur par défaut est : libre accès à tous les fichiers.

[7.1.1.12 ini.gpc-order](#)

[\[Notes en ligne\]](#)

gpc_order chaîne de caractères

- Etablit l'ordre de préséance des méthodes GET/POST/COOKIE. Par défaut, cette directive est établie à "GPC". En affectant "GP" à cette directive, PHP ignorera les cookies, et écrasera toute méthode GET utilisée par une méthode POST avec des variables du même nom.

[7.1.1.13 ini.ignore-user-abort](#)

[\[Notes en ligne\]](#)

ignore_user_abort chaîne de caractères

- Désactivée par défaut. Si cette directive est activée, alors tous les scripts lancés iront jusqu'à leur terme, même si le client se déconnecte en plein milieu. Voir aussi la fonction [ignore_user_abort\(\)](#).

[7.1.1.14 ini.include-path](#)

[\[Notes en ligne\]](#)

include_path string

- Spécifie une liste de dossier où les fonctions [require\(\)](#), [include\(\)](#) et `fopen_with_path` (NDtraducteur : cette fonction semble avoir disparue) iront chercher les fichiers. Le format est le même que celui de la variable d'environnement **PATH** : une liste de dossiers, séparés par des deux points (:) sous UNIX, et des points virgules (;), sous Windows.

UNIX include_path

`include_path=.: /home/httpd/php-lib`

Windows include_path

`include_path=".;c:\www\phplib"`

La valeur par défaut pour cette directive est ., c'est à dire le dossier courant.

[7.1.1.15 ini.isapi-ext](#)

[\[Notes en ligne\]](#)

isapi_ext string

- (Aucune documentation n'est fournie)

[7.1.1.16 ini.log-errors](#)

[\[Notes en ligne\]](#)

log_errors boolean

- Indique où les messages d'erreur générés doivent être écrits. Cette fonction est spécifique aux

serveurs.

[7.1.1.17 ini.magic-quotes-gpc](#)

[\[Notes en ligne\]](#)

magic_quotes_gpc boolean

- Fixe le mode *magic_quotes* pour les opérations GPC (Get/Post/Cookie). Lorsque *magic_quotes* est activé, tous les caractères ' (guillemets simples), " (guillemets doubles), \ (antislash) et NUL sont échappés avec un antislash. Si *magic_quotes_sybase* fonctionne aussi, les guillemets simples seront échappés avec un autre guillemet simple, plutôt qu'un antislash.

[7.1.1.18 ini.magic-quotes-runtime](#)

[\[Notes en ligne\]](#)

magic_quotes_runtime boolean

- Si *magic_quotes_runtime* est activé, toutes les fonctions qui retournent des données d'une source externe, y compris les bases de données et les fichiers texte, verront leur guillemets échappés avec un antislash. Si *magic_quotes_sybase* est aussi activé, les guillemets simples seront échappés avec un autre guillemet simple, plutôt qu'un antislash.

[7.1.1.19 ini.magic-quotes-sybase](#)

[\[Notes en ligne\]](#)

magic_quotes_sybase boolean

- Si *magic_quotes_sybase* est activé, les guillemets simples seront échappés avec un autre guillemet simple, plutôt qu'un antislash, si *magic_quotes_gpc* ou *magic_quotes_runtime* est activé.

[7.1.1.20 ini.max-execution-time](#)

[\[Notes en ligne\]](#)

max_execution_time integer

- Fixe le temps maximal d'exécution d'un script, en secondes. Cela permet d'éviter que des scripts en boucles infinies ne saturent le serveur.

[7.1.1.21 ini.memory-limit](#)

[\[Notes en ligne\]](#)

memory_limit entier

- Grâce à cette option, vous pouvez donner la quantité maximale de mémoire qu'un script peut allouer. Cela permet de réserver toute la mémoire d'un serveur à un seul script.

[7.1.1.22 ini.nsapi-ext](#)

[\[Notes en ligne\]](#)

nsapi_ext chaîne de caractères

- Aucune documentation n'est fournie.

[7.1.1.23 ini.register-globals](#)

[\[Notes en ligne\]](#)

register_globals boolean

- Cette option active l'enregistrement des variables EGPCS (Environnement, GET, POST, Cookie, Serveur), en tant que variables globales. Vous pouvez désactiver cette fonction si vous ne voulez pas truffer vos scripts avec des valeurs utilisateurs. Cette option est surtout utile lorsqu'elle est utilisée conjointement avec [7.1.1.27 ini.track-vars](#) – dans ce cas, vous pouvez accéder à toutes les variables EGPCS grâce aux tableaux *\$HTTP_ENV_VARS*, *\$HTTP_GET_VARS*, *\$HTTP_POST_VARS*, *\$HTTP_COOKIE_VARS*, et *\$HTTP_SERVER_VARS*.

[7.1.1.24 ini.short-open-tag](#)

[\[Notes en ligne\]](#)

short_open_tag booléen

- Active ou désactive l'utilisation des balises courtes, (<? ?>). Si vous voulez utiliser PHP et XML en même temps, vous devez désactiver cette option. Si cette option est désactivée, vous devez utiliser la forme longue des tags, (<?php ?>).

[7.1.1.25 ini.sql.safe-mode](#)

[\[Notes en ligne\]](#)

sql.safe_mode booléen

- Aucune documentation n'est fournie.

[7.1.1.26 ini.track-errors](#)

[\[Notes en ligne\]](#)

track_errors booléen

- Si cette option est activée, le dernier message d'erreur sera placé dans la variable globale *\$php_errormsg*.

[7.1.1.27 ini.track-vars](#)

[\[Notes en ligne\]](#)

track_vars booléen

- Si cette option est activée, lors de l'appel des méthodes GET, POST et l'utilisation des cookies, les variables sont disponibles dans des tableaux associatifs globaux appelés respectivement *\$HTTP_GET_VARS*, *\$HTTP_POST_VARS* et *\$HTTP_COOKIE_VARS*.

[7.1.1.28 ini.upload-tmp-dir](#)

[\[Notes en ligne\]](#)

upload_tmp_dir chaîne de caractères

- Indique le répertoire utilisé lors du chargement d'un fichier sur un serveur. Ce répertoire doit être accessible en lecture pour l'utilisateur qui lance le script PHP.

[7.1.1.29 ini.user-dir](#)

[\[Notes en ligne\]](#)

user_dir chaîne de caractères

- Répertoire où sont stockés les fichiers PHP dans le répertoire d'un utilisateur. Par exemple, public_html.

[7.1.1.30 ini.warn-plus-overloading](#)

[\[Notes en ligne\]](#)

warn_plus_overloading booléen

- Si cette option est activée, PHP émet un warning lorsque l'opérateur plus (+) est utilisé sur une chaîne de caractères. Cela permet de repérer plus facilement les scripts qui doivent être réécrits en utilisant l'opérateur de concaténation (.) plutôt que l'opérateur plus.

[7.1.2 Configuration des directives concernant le mail](#)

[\[Notes en ligne\]](#)

[7.1.2.1 ini.smtp](#)

[\[Notes en ligne\]](#)

SMTP chaîne de caractères

- Sous Windows, adresse IP ou nom que PHP doit utiliser pour envoyer du mail avec la fonction [mail\(\)](#).

[7.1.2.2 ini.sendmail-from](#)

[\[Notes en ligne\]](#)

sendmail_from chaîne de caractères

- Sous Windows, valeur du champs "From:" qui doit être utilisée lors de l'envoi de mail.

[7.1.2.3 ini.sendmail-path](#)

[\[Notes en ligne\]](#)

sendmail_path chaîne de caractères

- Localisation du programme de `sendmail`, habituellement `/usr/sbin/sendmail` ou `/usr/lib/sendmail`. configure essaye de repérer la présence de `sendmail` par lui même, et affecte ce résultat par défaut. En cas de problème de localisation, vous pouvez établir une nouvelle valeur par défaut ici.

Tout système n'utilisant pas `sendmail` doit établir cette directive à la valeur chemin du programme de substitution qui remplace le serveur de mail, si celui-ci existe, par exemple, [Qmail](#). Dans ce cas la, vous devez mettre: `/var/qmail/bin/sendmail`.

7.1.3 Directives de configuration du "Safe Mode"

[\[Notes en ligne\]](#)

7.1.3.1 ini.safe-mode

[\[Notes en ligne\]](#)

safe_mode booléen

- Cette directive active ou désactive l'option "safe mode". Lisez le chapitre [6 Sécurité](#) pour plus d'informations.

7.1.3.2 ini.safe-mode-exec-dir

[\[Notes en ligne\]](#)

safe_mode_exec_dir chaîne de caractères

- Si l'option "SAFE MODE" est activée, [system\(\)](#) et les autres fonctions exécutant des programmes systèmes refusent de se lancer si ces programmes ne sont pas placés dans ce répertoire.

7.1.4 Directives de configuration de débbugage.

[\[Notes en ligne\]](#)

7.1.4.1 ini.debugger.host

[\[Notes en ligne\]](#)

debugger.host chaîne de caractères

- Adresse IP ou nom de l'hôte utilisé pour le débogage.

7.1.4.2 ini.debugger.port

[\[Notes en ligne\]](#)

debugger.port chaîne de caractères

- Numéro du port utilisé pour le débogage.

7.1.4.3 ini.debugger.enabled

[\[Notes en ligne\]](#)

debugger.enabled booléen

- Activation ou désactivation du debugger.

7.1.5 Directives de chargement des extensions

[\[Notes en ligne\]](#)

[7.1.5.1 ini.enable-dl](#)

[\[Notes en ligne\]](#)

enable_dl booléen

- Cette directive n'est réellement utile que dans le cas d'une compilation comme module Apache. Vous pouvez activer le chargement dynamique des extensions avec la fonction [dl\(\)](#), et cela de manière locale à chaque serveur virtuel ou à chaque répertoire.

La principale raison qui pousse à désactiver le chargement dynamique est un problème de sécurité.

Lorsque le chargement dynamique est activé, il est possible d'ignorer les directives [7.1.3.1 ini.safe-mode](#) ou "open_basedir".

Par défaut, il est possible d'utiliser le chargement dynamique, sauf lorsque la directive [7.1.3.1 ini.safe-mode](#) est activée. En effet, il est alors impossible d'utiliser la fonction [dl\(\)](#).

[7.1.5.2 ini.extension-dir](#)

[\[Notes en ligne\]](#)

extension_dir chaîne de caractères

- Définit le répertoire dans lequel le PHP doit chercher les extensions lors du chargement dynamique.

[7.1.5.3 ini.extension](#)

[\[Notes en ligne\]](#)

extension chaîne de caractères

- Définit les extensions qui doivent être chargées lors du démarrage du PHP.

[7.1.6 MySQL Configuration Directives](#)

[\[Notes en ligne\]](#)

[7.1.6.1 ini.mysql.allow-persistent](#)

[\[Notes en ligne\]](#)

mysql.allow_persistent booléen

- Active ou désactive les connexions persistentes à la base de données MySQL.

[7.1.6.2 ini.mysql.default-host](#)

[\[Notes en ligne\]](#)

mysql.default_host chaîne de caractères

- Adresse par défaut du serveur, à utiliser lors de la connexion à un serveur MySQL, si aucun hôte n'est spécifié.

[7.1.6.3 ini.mysql.default-user](#)

[\[Notes en ligne\]](#)

mysql.default_user chaîne de caractères

- Utilisateur par défaut, à utiliser lors de la connexion à un serveur MySQL, si aucun utilisateur n'est spécifié.

[7.1.6.4 ini.mysql.default-password](#)

[\[Notes en ligne\]](#)

mysql.default_password chaîne de caractères

- Mot de passe par défaut, à utiliser lors de la connexion à un serveur MySQL, si aucun mot de passe n'est spécifié.

[7.1.6.5 ini.mysql.max-persistent](#)

[\[Notes en ligne\]](#)

mysql.max_persistent entier

- Nombre maximum de connexions persistantes à une base de donnée MySQL, par processus.

[7.1.6.6 ini.mysql.max-links](#)

[\[Notes en ligne\]](#)

mysql.max_links entier

- Nombre de connexion maximum à une base de donnée MySQL, par processus, incluant les connexions persistantes

[7.1.7 Directives de configuration mSQL](#)

[\[Notes en ligne\]](#)

[7.1.7.1 ini.msql.allow-persistent](#)

[\[Notes en ligne\]](#)

msql.allow_persistent booléen

- Active ou désactive les connexions persistentes à la base de données mSQL.

[7.1.7.2 ini.msql.max-persistent](#)

[\[Notes en ligne\]](#)

msql.max_persistent entier

- Nombre maximum de connexions persistantes à une base de donnée mSQL, par processus.

[7.1.7.3 ini.msql.max-links](#)

[\[Notes en ligne\]](#)

msql.max_links entier

- Nombre maximum de connexions à une base de donnée mSQL, par processus, incluant les connexions persistantes.

7.1.8 Directives de configuration Postgres

[\[Notes en ligne\]](#)

7.1.8.1 ini.pgsql.allow-persistent

[\[Notes en ligne\]](#)

pgsql.allow_persistent booléen

- Active ou désactive les connexions persistantes à la base de données Postgres.

7.1.8.2 ini.pgsql.max-persistent

[\[Notes en ligne\]](#)

pgsql.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Postgres, par processus.

7.1.8.3 ini.pgsql.max-links

[\[Notes en ligne\]](#)

pgsql.max_links entier

- Nombre maximal de connexions à une base de donnée Postgres, par processus, incluant les connexions persistantes.

7.1.9 Directives de configuration SESAM

[\[Notes en ligne\]](#)

7.1.9.1 ini.sesam-oml

[\[Notes en ligne\]](#)

sesam_oml string

- Nom de la librairie BS2000 PLAM contenant les pilotes de connexion SESAM. Obligatoire pour les fonctions SESAM. La librairie BS2000 PLAM doit être configurée avec ACCESS=READ,SHARE=YES car elle doit être accessible à l'utilisateur Apache.

7.1.9.2 ini.sesam-configfile

[\[Notes en ligne\]](#)

sesam_configfile string

- Nom du fichier de configuration de l'application SESAM. Obligatoire pour les fonctions SESAM. Le fichier BS2000 doit être accessible à l'utilisateur Apache. Le fichier de configuration de l'application va contenir la configuration sur le schéma suivant (voir le manuel SESAM) :

```
CNF=B
NAM=K
NOTYPE
```


[7.1.9.3 ini.sesam-messagecatalog](#)

[\[Notes en ligne\]](#)

sesam_messagecatalog string

- Nom du catalogue de messages SESAM. Dans la plus part des cas, cette directive n'est pas nécessaire. Seulement, si le fichier de messages SESAM n'est pas installé dans la table de messages BS2000, il faut indiquer sa localisation avec cette directive.
Le catalogue de messages doit être paramétré avec ACCESS=READ,SHARE=YES car elle doit être accessible à l'utilisateur Apache.

[7.1.10 Directives de configuration Sybase](#)

[\[Notes en ligne\]](#)

[7.1.10.1 ini.sybase.allow-persistent](#)

[\[Notes en ligne\]](#)

sybase.allow_persistent booléen

- Active ou désactive les connexions persistentes à la base de données Sybase.

[7.1.10.2 ini.sybase.max-persistent](#)

[\[Notes en ligne\]](#)

sybase.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Sybase par processus.

[7.1.10.3 ini.sybase.max-links](#)

[\[Notes en ligne\]](#)

sybase.max_links entier

- Nombre maximum de connexions à une base de données Sybase, par processus, incluant les connexions persistantes.

[7.1.11 Sybase-CT Configuration Directives](#)

[\[Notes en ligne\]](#)

[7.1.11.1 ini.sybct.allow-persistent](#)

[\[Notes en ligne\]](#)

sybct.allow_persistent booléen

- Active ou désactive les connexions persistantes à la base de données Sybase-CT. Par défaut, cette option est activée.

[7.1.11.2 ini.sybct.max-persistent](#)

[\[Notes en ligne\]](#)

sybct.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Sybase-CT par processus. Par défaut, cette option est à -1, ce qui signifie que le nombre de connexion est illimité.

[7.1.11.3 ini.sybct.max-links](#)

[\[Notes en ligne\]](#)

sybct.max_links entier

- Nombre maximum de connexions à une base de données Sybase-CT, par processus, incluant les connexions persistantes. Par défaut, cette option est à -1, ce qui signifie que le nombre de connexions est illimitée.

[7.1.11.4 ini.sybct.min-server-severity](#)

[\[Notes en ligne\]](#)

sybct.min_server_severity entier

- Les messages en provenance du serveur d'un niveau d'erreur égal à *sybct.min_server_severity* seront considérés comme des alertes (warnings). Cette valeur peut être modifiée à l'intérieur du script en appelant la fonction *sybase_min_server_severity* (NDtraducteur : cette fonction semble ne pas exister). Par défaut, cette valeur vaut 10.

[7.1.11.5 ini.sybct.min-client-severity](#)

[\[Notes en ligne\]](#)

sybct.min_client_severity entier

- Les messages en provenance de la librairie client avec un niveau d'erreur égal ou supérieur à *sybct.min_client_severity* seront considérés comme des alertes. Cette valeur peut être modifiée à l'intérieur du script en appelant la fonction *sybase_min_client_severity* (NDtraducteur : cette fonction semble ne pas exister). Par défaut, cette valeur vaut 10, ce qui annule tout rapport d'erreur.

[7.1.11.6 ini.sybct.login-timeout](#)

[\[Notes en ligne\]](#)

sybct.login_timeout entier

- Délai de validité d'une tentative de connexion. Il est à noter que si *max_execution_time* est dépassé avant que la connexion n'expire, le script sera terminé avant le message d'erreur. Par défaut, cette valeur vaut 1 minute.

[7.1.11.7 ini.sybct.timeout](#)

[\[Notes en ligne\]](#)

sybct.timeout entier

- Temps maximum en secondes avant qu'une tentative de requête "select_db" ou "query" non aboutie renvoie une erreur. Il est à noter que si max_execution_time est dépassé avant que la requête n'expire, votre script sera terminé avant le message d'erreur. Par défaut, il n'y a pas de limite.

[7.1.11.8 ini.sybct.hostname](#)

[\[Notes en ligne\]](#)

sybct.hostname chaîne de caractères

- Nom de l'hôte à partir duquel vous vous connectez, afin d'être affiché par la fonction sp_who (NDtraducteur : cette fonction semble ne pas exister). Par défaut, cette valeur égale à 0.

[7.1.12 Directives de configuration Informix](#)

[\[Notes en ligne\]](#)

[7.1.12.1 ini.ifx.allow-persistent](#)

[\[Notes en ligne\]](#)

ifx.allow_persistent booléen

- Active les connexions persistantes à une base de données Informix.

[7.1.12.2 ini.ifx.max-persistent](#)

[\[Notes en ligne\]](#)

ifx.max_persistent entier

- Nombre maximum de connexions persistantes à une base de données Informix, par processus.

[7.1.12.3 ini.ifx.max-links](#)

[\[Notes en ligne\]](#)

ifx.max_links entier

- Nombre maximum de connexions à une base de données Informix par processus, en incluant les connexions persistantes.

[7.1.12.4 ini.ifx.default-host](#)

[\[Notes en ligne\]](#)

ifx.default_host chaîne de caractères

- Hôte par défaut où se connecter si aucun hôte n'est spécifié par les fonctions [ifx_connect\(\)](#) ou [ifx_pconnect\(\)](#).

[7.1.12.5 ini.ifx.default-user](#)

[\[Notes en ligne\]](#)

ifx.default_user chaîne de caractères

- Utilisateur par défaut si aucun utilisateur n'est spécifié par les fonctions [ifx_connect\(\)](#) ou [ifx_pconnect\(\)](#).

[7.1.12.6 ini.ifx.default-password](#)

[\[Notes en ligne\]](#)

ifx.default_password chaîne de caractères

- Mot de passe par défaut si aucun mot de passe n'est spécifié par les fonctions [ifx_connect\(\)](#) ou [ifx_pconnect\(\)](#).

[7.1.12.7 ini.ifx.blobinfile](#)

[\[Notes en ligne\]](#)

ifx.blobinfile booléen

- Lorsque cette option est activée, les colonnes de type "blob" seront retournées dans un fichier. Par défaut, elles seront retournées en mémoire. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_blobinfile_mode\(\)](#).

[7.1.12.8 ini.ifx.textasvarchar](#)

[\[Notes en ligne\]](#)

ifx.textasvarchar booléen

- Lorsque cette option est activée, les colonnes de type "TEXT" seront retournées dans une chaîne de caractères. Par défaut, elles seront retournées en mémoire. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_textasvarchar\(\)](#).

[7.1.12.9 ini.ifx.byteasvarchar](#)

[\[Notes en ligne\]](#)

ifx.byteasvarchar booléen

- Lorsque cette option est activée, les colonnes de type "BYTE" seront retournées dans une chaîne de caractères. Par défaut, elles seront retournées en mémoire. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_textasvarchar\(\)](#).

[7.1.12.10 ini.ifx.charasvarchar](#)

[\[Notes en ligne\]](#)

ifx.charasvarchar booléen

- Lorsque cette option est activée, les espaces en fin de chaîne de caractères seront conservés lors d'une commande FETCH.

[7.1.12.11 ini.ifx.nullformat](#)

[\[Notes en ligne\]](#)

ifx.nullformat booléen

- Lorsque cette option est activée, les colonnes de valeur NULL seront retournées comme des chaînes de caractères vides. Il est possible de modifier dynamiquement cette valeur grâce à la fonction [ifx_nullformat\(\)](#).

7.1.13 Directives de configuration pour les calculs mathématiques.

[\[Notes en ligne\]](#)

7.1.13.1 ini.bcmath.scale

[\[Notes en ligne\]](#)

bcmath.scale entier

- Nombre de chiffres après la virgule pour toutes les fonctions de précision mathématique arbitraire.

7.1.14 Directives de configuration du navigateur.

[\[Notes en ligne\]](#)

7.1.14.1 ini.browscap

[\[Notes en ligne\]](#)

browscap chaîne de caractères

- Nom du fichier de descriptif des clients HTML. Voir aussi [get_browser\(\)](#).

7.1.15 Directives de configuration du driver ODBC unifié

[\[Notes en ligne\]](#)

7.1.15.1 ini.uodbc.default-db

[\[Notes en ligne\]](#)

uodbc.default_db chaîne de caractères

- Source de données ODBC à utiliser par défaut avec les fonctions [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

7.1.15.2 ini.uodbc.default-user

[\[Notes en ligne\]](#)

uodbc.default_user chaîne de caractères

- Nom d'utilisateur défaut avec les fonctions [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

7.1.15.3 ini.uodbc.default-pw

[\[Notes en ligne\]](#)

uodbc.default_pw chaîne de caractères

- Mot de passe par défaut dans les fonctions [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

[7.1.15.4 ini.uodbc.allow-persistent](#)

[\[Notes en ligne\]](#)

uodbc.allow_persistent booléen

- Cette option active ou désactive les connexions persistantes à la base de données, via le canal ODBC.

[7.1.15.5 ini.uodbc.max-persistent](#)

[\[Notes en ligne\]](#)

uodbc.max_persistent entier

- Nombre maximum de connexions persistantes autorisées à la base de données.

[7.1.15.6 ini.uodbc.max-links](#)

[\[Notes en ligne\]](#)

uodbc.max_links entier

- Nombre maximum de connexions (persistantes ou non), par processus, à la base de données.

8 Caractéristiques

[\[Notes en ligne\]](#)

8.1 Gestion des connexions

[\[Notes en ligne\]](#)

Note : *Les informations suivantes ne sont valables qu'à partir de la version 3.0.7.*

Le statut des connexions est conservé en interne par PHP. Il y a trois états possibles :

- 0 – NORMAL (normal)
- 1 – ABORTED (annulé)
- 2 – TIMEOUT (périmé)

Lorsqu'un script PHP est en cours d'exécution, son état est NORMAL. Si le client distant se déconnecte, le statut devient ABORTED. En général, une telle déconnexion provient d'un arrêt temporaire. Si la durée maximale d'exécution de PHP est dépassée, (voir [set_time_limit\(\)](#)), le script prend le statut TIMEOUT. Vous pouvez en outre, décider si vous voulez que la déconnexion d'un client provoque l'arrêt de votre script. Il est parfois pratique de terminer le script, même si le client n'est plus là pour recevoir les informations. Cependant, par défaut, le script sera interrompu, et terminé dès que le client se déconnecte. Ce comportement peut être modifié avec la directive `ignore_user_abort` dans le fichier ``php.ini`` ou bien avec la directive Apache `ignore_user_abort` du fichier Apache ``httpd.conf`` ou avec la fonction [ignore_user_abort\(\)](#). Si vous ne demandez pas à PHP d'ignorer la déconnexion, et que l'utilisateur se déconnecte, le script sera terminé. La seule exception est si vous avez enregistré une fonction de fermeture, avec [register_shutdown_function\(\)](#). Avec une telle fonction, lorsque l'utilisateur interrompt sa requête, à la prochaine exécution du script, PHP va s'apercevoir que le dernier script n'a pas été terminé, et il va déclencher la fonction de fermeture. Cette fonction sera aussi appelée à la fin du script, si celui-ci se termine normalement. Pour pouvoir avoir un comportement différent suivant l'état du script lors de sa finalisation, vous pouvez exécuter des commandes spécifiques à la déconnexion grâce à la commande [connection_aborted\(\)](#). Cette fonction retournera TRUE si la connexion a été annulée. Votre script peut aussi expirer après un laps de temps. Par défaut, le délai est de 30 secondes. Cette valeur peut être changée en utilisant la directive PHP `max_execution_time` dans le fichier ``php.ini`` ou avec la directive `php3_max_execution_time`, dans le fichier Apache ``httpd.conf`` ou encore avec la fonction [set_time_limit\(\)](#). Lorsque le délai expire, le script est terminé, et comme pour la déconnexion du client, une fonction de finalisation sera appelée. Dans cette fonction, vous pouvez savoir si c'est le délai d'expiration qui a causé la fin du script, en appelant la fonction [connection_timeout\(\)](#). Cette fonction retournera vrai si le délai d'expiration a été dépassé.

Une chose à noter et que les deux cas ABORTED et TIMEOUT peuvent être appelés en même temps. Ceci est possible si vous demandez à PHP d'ignorer les déconnexions des utilisateurs. PHP va quand même noter le fait que l'utilisateur s'est déconnecté, mais le script va continuer. Puis, lorsqu'il atteint la limite de temps, le script va expirer. A ce moment là, les deux fonctions [connection_timeout\(\)](#) et [connection_aborted\(\)](#) vont retourner TRUE. Vous pouvez aussi vérifier les deux états en un seul appel avec la fonction [connection_status\(\)](#). Cette fonction va retourner un champs de bits, avec les états. Si les deux états sont actifs, l'état retourné prendra la valeur 3.

8.2 Cookies

[\[Notes en ligne\]](#)

PHP supporte les cookies de manière transparente. Les cookies sont un mécanisme d'enregistrement d'informations sur le client, et de lecture de ces informations. Ce système permet d'authentifier et de suivre les visiteurs. Vous pouvez envoyer un cookie avec la commande [setcookie\(\)](#). Les cookies font partie de l'entête **HTTP**, ce qui impose que [setcookie\(\)](#) soit appelé avant tout affichage sur le client. Ce sont les mêmes limitations que pour [header\(\)](#).

Tous les cookies qui sont envoyés au client seront automatiquement retournés au script PHP, et transformés en variable, exactement comme pour GET et POST. Si vous souhaitez affecter plusieurs valeurs à un seul cookie, ajoutez[] au nom du cookie. Pour plus details, reportez vous à la fonction [setcookie\(\)](#).

8.3 Gestion des erreurs

[\[Notes en ligne\]](#)

Il y a plusieurs types d'erreur et d'alerte.

Valeur	Constante	Description	Note
1	E_ERROR	Erreur fatale d'exécution	@tab
2	E_WARNING	Alerte d'exécution (erreur non-fatale)	@tab
4	E_PARSE	Erreur de compilation	@tab
8	E_NOTICE	Notes d'exécution (moins critique que les alertes) @tab	@tab
16	E_CORE_ERROR	Erreurs qui surviennent lors de l'initialisation de PHP	PHP 4 seulement
32	E_CORE_WARNING	Alertes qui surviennent lors de l'initialisation de PHP @tab PHP 4 seulement	
64	E_COMPILE_ERROR	Erreur fatale de compilation	PHP 4 seulement
128	E_COMPILE_WARNING	Alerte de compilation (erreur non fatale)	PHP 4 seulement
256	E_USER_ERROR	Erreur générée par l'utilisateur	PHP 4 seulement
512	E_USER_WARNING	Alerte générée par l'utilisateur	PHP 4 seulement
1024	E_USER_NOTICE	Note générée par l'utilisateur	PHP 4 seulement
@tab E_ALL	Toutes les erreurs ci dessus	@tab	

Les valeurs ci-dessus (numériques ou symbolique) sont utilisée pour construire un champs de bit, qui spécifie quelles erreurs rapporter. Vous pouvez utiliser les [9.7.3 Bitwise Operators](#) pour combiner ces valeurs

et masquer uniquement celle qui vous interesse Notez que seuls, '|', '~', '!', et '&' seront utilisables dans `php.ini`, et qu'aucun opérateur ne sera utilisable dans `php3.ini`.

En PHP 4, la valeur par défaut de [7.1.1.10 ini.error-reporting](#) est à E_ALL & ~E_NOTICE, ce qui signifie que toutes les erreurs et alertes seront affichées, mais pas les notes. En PHP 3, la valeur par défaut est (E_ERROR | E_WARNING | E_PARSE), c'est à dire la même chose. Notez bien que ces constantes ne sont pas supportées dans le fichier `php3.ini` de PHP 3, la valeur de [7.1.1.10 ini.error-reporting](#) doit être numériques, c'est à dire 7.

La valeur initiale peut être modifiée dans le fichier .ini, avec la directive [7.1.1.10 ini.error-reporting](#), dans le fichier de configuration d'Apache `httpd.conf`, avec la directive `php_error_reporting` (`php3_error_reporting` pour PHP 3), et enfin, dans le script même, en utilisant la fonction [error_reporting\(\)](#). Lorsque vous portez votre code ou vos serveurs de PHP 3 en PHP 4 vous devez vérifier les options et les appels à [error_reporting\(\)](#). Sinon, vous courez le risque d'inactiver certains types d'erreurs et notamment E_COMPILE_ERROR. Cela peut conduire à des documents vides, sans aucun retour d'erreur.

Toutes les [9.4 Les expressions](#) peuvent être appelée avec le préfixe "@", qui annule le rapport d'erreur pour cette expression en particulier. Si une erreur survient durant une telle expression, et que l'option de [7.1.1.26 ini.track-errors](#) est activée, vous pourrez trouver le message d'erreur dans la variable globale,

\$php_errormsg.

Note : Le préfixe opérateur [9.7.5 Opérateur de contrôle d'erreur](#) ne supprimera pas les messages liés aux erreurs d'analyse.

Actuellement, le préfixe [9.7.5 Opérateur de contrôle d'erreur](#), opérateur de rapport d'erreur désactive tous les rapports, y compris les erreurs critiques qui interrompent le script. Entre autre, cela signifie que si vous utilisez [9.7.5 Opérateur de contrôle d'erreur](#) pour supprimer des erreurs dans une fonction qui n'existe pas, ou qui a été mal orthographiée, le script sera terminé sans aucune indication.

Ci dessous, voici un exemple de gestion des erreurs avec PHP. On définit une fonction de gestion des erreurs qui enregistre les informations dans un fichier (au format XML), et email le développeur en cas d'erreur critique.

Utiliser le contrôle d'erreur dans un script

```
<?php
// Nous effectuons nous même notre contrôle d'erreur.
error_reporting(0);
// Fonction de gestion des erreurs utilisateur
function usererrorhandler($errno, $errormsg, $filename, $linenum, $vars) {
    // timestamp pour dater l'erreur
    $dt = date("Y-m-d H:i:s (T)");
    // definit un tableau associatif avec les chaînes d'erreur
    // en réalité, les seules entrées que nous considérerons
    // seront 2,8,256,512 et 1024
    $errortype = array(
1 => "Erreur",
2 => "Alerte",
4 => "Erreur d'analyse",
8 => "Note",
16 => "Erreur interne",
32 => "Alerte interne",
64 => "Erreur de compilation",
128 => "Alerte de compilation",
256 => "Erreur utilisateur",
512 => "Alerte utilisateur",
1024=> "Note utilisateur"
    );
    // ensemble d'erreur pour lesquelles une trace sera conservée
    $user_errors = array(E_USER_ERROR, E_USER_WARNING, E_USER_NOTICE);
    $err = "<errorentry>\n";
    $err .= "\t<datetime>".$dt."</datetime>\n";
    $err .= "\t<errornum>".$errno."</errnumber>\n";
```

```

    $err .= "\t<errortype>".$errortype[$errno]."</errortype>\n";
    $err .= "\t<errmsg>".$errmsg."</errmsg>\n";
    $err .= "\t<scriptname>".$filename."</scriptname>\n";
    $err .= "\t<scriptlinenum>".$linenum."</scriptlinenum>\n";
if (in_array($errno, $user_errors))
    $err .= "\t<vartrace>".wddx_serialize_value($vars,"Variables")."</vartrace>\n";
    $err .= "</errorentry>\n\n";
    // pour test
    // echo $err;
    // sauve l'erreur dans le fichier, et email moi si l'erreur est critique
error_log($err, 3, "/usr/local/php4/error.log");
if ($errno == E_USER_ERROR)
mail("phpdev@mydomain.com","Critical User Error",$err);
}
function distance($vect1, $vect2) {
if (!is_array($vect1) || !is_array($vect2)) {
trigger_error("Paramètres incorrects : arrays attendus", E_USER_ERROR);
return NULL;
}
if (count($vect1) != count($vect2)) {
trigger_error("Les vecteurs doivent être de la même taille", E_USER_ERROR);
return NULL;
}
for ($i=0; $i<count($vect1); $i++) {
    $c1 = $vect1[$i]; $c2 = $vect2[$i];
    $d = 0.0;
if (!is_numeric($c1)) {
trigger_error("La coordonnée $i du vecteur 1 n'est pas un nombre. Remplacée par zéro",
E_USER_WARNING);
    $c1 = 0.0;
}
if (!is_numeric($c2)) {
trigger_error("La coordonnée $i du vecteur 2 n'est pas un nombre. Remplacée par zéro",
E_USER_WARNING);
    $c2 = 0.0;
}
    $d += $c2*$c2 - $c1*$c1;
}
return sqrt($d);
}
$old_error_handler = set_error_handler("userErrorHandler");
// Constante indéfinie, génère une alerte
$t = I_AM_NOT_DEFINED;
// définition de quelques "vecteurs"
$a = array(2,3,"bla");
$b = array(5.5, 4.3, -1.6);
$c = array(1,-3);
// génère une erreur utilisateur
$t1 = distance($c,$b)."\n";
// génère une autre erreur utilisateur
$t2 = distance($b,"i am not an array")."\n";
// génère une alerte
$t3 = distance($a,$b)."\n";
?>

```

Ceci est un exemple simple, qui montre comment utiliser les fonctions de [10.18 Gestion des erreurs](#). Voir aussi [error_reporting\(\)](#), [error_log\(\)](#), [set_error_handler\(\)](#), [restore_error_handler\(\)](#), [trigger_error\(\)](#), [user_error\(\)](#)

8.4 Gestion des chargements de fichier

[\[Notes en ligne\]](#)

8.4.1 Chargements de fichiers par méthode POST

[\[Notes en ligne\]](#)

PHP est capable de recevoir des fichiers émis par un navigateur conforme à la norme RFC-1867 (c'est à dire Netscape Navigator 3 ou supérieur, Microsoft Internet Explorer 3 avec un patch de Microsoft, ou supérieur sans le patch). Cette fonctionnalité permet de charger des fichiers texte binaire. Avec l'authentification et les fonctions de manipulation des fichiers, vous avez un contrôle total sur le chargement et la gestion des fichiers chargés.

Notez bien que PHP supporte aussi le chargement par la méthode PUT comme dans le navigateur Netscape Composer et les clients Amaya du W3C. Reportez vous au chapitre sur le [8.4.4 Chargement par méthode PUT](#).

Un écran de chargement de fichiers peut être constitué en créant un formulaire de la manière suivante :

Formulaire de chargement de fichier

```
<FORM ENCTYPE="multipart/form-data" ACTION="_URL_" METHOD=POST>
<INPUT TYPE="hidden" name="MAX_FILE_SIZE" value="1000">
Send this file: <INPUT NAME="userfile" TYPE="file">
<INPUT TYPE="submit" VALUE="Send File">
</FORM>
```

Le paramètre `_URL_` doit pointer sur un fichier PHP. L'option `MAX_FILE_SIZE` cachée doit précéder le nom du fichier à charger, et représente la taille maximale du fichier à charger. La valeur est donnée en octets.

Dans ce script, les valeurs suivantes doivent être définies pour assurer un chargement correct :

En PHP 3, les variables suivantes seront définies dans le script de destination, en cas de téléchargement réussi, et en supposant que [7.1.1.23 ini.register-globals](#) est activé dans le fichier ``php.ini'`. Si [7.1.1.27 ini.track-vars](#) est activé, elles seront aussi disponibles dans le dossier `$HTTP_POST_VARS`. Notez que les noms des variables suivantes supposent que nom du fichier téléchargé est 'userfile', comme présenté dans l'exemple ci-dessus.

- ***\$userfile*** – Le nom temporaire du fichier qui sera chargé sur la machine serveur.
- ***\$userfile_name*** – Le nom du fichier original sur le système de l'expéditeur.
- ***\$userfile_size*** – La taille du fichier envoyé en octets.
- ***\$userfile_type*** – Le type MIME du fichier, si le navigateur a fourni cette information. Par exemple, "image/gif".

Notez que "\$userfile" prend la valeur qui est passée dans le champs INPUT de type TYPE=file. Dans l'exemple ci dessus, nous avons choisi de l'appeler "userfile".

En PHP 4, le comportement est légèrement différent, car c'est la variable d'environnement `$HTTP_POST_FILES`, qui contiendra les informations sur les fichiers téléchargés. Ces informations sont disponibles dans si [7.1.1.27 ini.track-vars](#) est activé, mais [7.1.1.27 ini.track-vars](#) est toujours activé dans les versions de PHP supérieure à la version 4.0.2.

Le contenu du tableau `$HTTP_POST_FILES` décrit ci dessous. Notez que l'on suppose ici que le nom du fichier téléchargé est 'userfile', comme présenté dans l'exemple ci-dessus :

\$HTTP_POST_FILES['userfile']['name']

- Le nom du fichier original sur la machine source.

`$HTTP_POST_FILES['userfile']['type']`

- Le type MIME du fichier, si le navigateur a fourni cette information. Par exemple, "image/gif".

`$HTTP_POST_FILES['userfile']['size']`

- La taille du fichier envoyé, en octets.

`$HTTP_POST_FILES['userfile']['tmp_name']`

- Le nom temporaire du fichier qui sera chargé sur la machine serveur.

Les fichiers seront enregistrés par défaut dans le dossier des fichiers temporaires, à moins qu'un autre dossier n'ait été fourni avec la directive de configuration [7.1.1.28 ini.upload-tmp-dir](#) du fichier ``php.ini'`. Le dossier par défaut du serveur peut être modifié grâce à la variable d'environnement **`TMPDIR`**, de l'utilisateur qui exécute PHP. Sa modification avec [putenv\(\)](#) depuis un script PHP ne fonctionnera pas. Cette variable d'environnement peut aussi être utilisée pour s'assurer que d'autres opérations fonctionnent avec les fichiers téléchargés.

Validation de fichiers téléchargés

Les exemples suivants fonctionnent sur les versions de PHP 3 supérieure à la version 3.0.16, et supérieure à la version 4.0.2 pour PHP 4. Reportez vous à la section des fonctions pour étudier [is_uploaded_file\(\)](#) et [move_uploaded_file\(\)](#).

```
38<lt?php
if (is_uploaded_file($userfile)) {
copy($userfile, "/dossier/des/fichiers/telecharges/");
} else {
echo "Attaque potentielle par fichier téléchargé : fichier '$userfile.'";
}
/* ...ou... */
move_uploaded_file($userfile, "/dossier/des/fichiers/telecharges");
?>
```

Pour les versions plus anciennes de PHP, vous devrez faire quelques chose comme : *Note : Cela ne fonctionnera PAS avec les versions de PHP 4 supérieure à 4.0.2. Cela repose sur des fonctionnalités internes à PHP qui on évoluée après cette version.*

```
38<lt?php
/* Test du fichier téléchargé. */
function is_uploaded_file($filename) {
if (!$tmp_file = get_cfg_var('upload_tmp_dir')) {
$tmp_file = dirname(tempnam('', ''));
}
$tmp_file .= '/' . basename($filename);
/* L'utilisateur peut avoir un slash final dans `php.ini'... */
return (ereg_replace('/+', '/', $tmp_file) == $filename);
}
if (is_uploaded_file($userfile)) {
copy($userfile, "/place/to/put/uploaded/file");
} else {
echo "Attaque potentielle par fichier téléchargé : fichier '$userfile.'";
}
?>
```

Le script PHP qui reçoit le fichier chargé doit pouvoir gérer le fichier de manière appropriée. Vous pouvez utiliser la variable `$file_size` pour recalculer tous les fichiers qui sont trop gros ou trop petit. Vous pouvez utiliser la variable `$file_type` pour recalculer les fichiers qui n'ont pas le bon type. Quelque soit les actions, ce script doit pouvoir supprimer le fichier du dossier temporaire, ou le déplacer ailleurs.

Le fichier sera automatiquement effacé du dossier temporaire à la fin du script, si il n'a pas été déplacé ou renommé.

8.4.2 Erreurs classiques

[\[Notes en ligne\]](#)

La variable `MAX_FILE_SIZE` ne peut pas spécifier une taille de fichier plus grande que la taille qui a été fixée par `upload_max_filesize`, dans le fichier ``php3.ini``, ou par `php3_upload_max_filesize` dans les directives Apache. La valeur par défaut est 2 Megaoctets.

Ne pas valider les fichiers que vous manipulez peut donner l'accès aux utilisateurs à des fichiers sensibles dans d'autres dossiers!

Attention : il semble que CERN httpd supprime tout ce qui est après le premier caractère dans l'entête MIME. Tant que c'est le cas, CERN httpd ne pourra pas effectuer de chargement.

8.4.3 Chargement multiples de fichiers

[\[Notes en ligne\]](#)

Il est possible de charger plusieurs fichiers en même temps, et recevoir les informations adéquates organisées sous forme de tableau. Pour ce faire, il faut utiliser la même syntaxe d'envoi dans le code HTML que pour les sélections ou boîte à cocher multiples.

Note : *Le support du chargement multiple de fichier a été ajouté dans la version 3.0.10.*

Chargement multiple de fichier

```
<form action="file-upload.html" method="post" enctype="multipart/form-data">
Send these files:<br>
  <input name="userfile[]" type="file"><br>
  <input name="userfile[]" type="file"><br>
  <input type="submit" value="Send files">
</form>
```

Lorsque le formulaire ci dessus est envoyé, les tableaux `$userfile`, `$userfile_name`, et `$userfile_size` seront initialisés (ainsi que `$HTTP_POST_VARS`). Chaque tableau sera de type numérique, et contiendra les valeurs appropriées pour le chargement des fichiers.

Par exemple, supposons que les noms de fichier ``/home/test/review.html`` et ``/home/test/xwp.out`` soient envoyés. Sans ce cas, `$userfile_name[0]` va contenir `review.html`, et `$userfile_name[1]` contiendra `xwp.out`. Similairement, `$userfile_size[0]` contiendra la taille de ``review.html``, etc...
`$userfile['name'][0]`, `$userfile['tmp_name'][0]`, `$userfile['size'][0]`, et `$userfile['type'][0]` sont aussi affectés.

8.4.4 Chargement par méthode PUT

[\[Notes en ligne\]](#)

PHP supporte la méthode HTTP PUT utilisée par les navigateurs tels que Netscape Composer et W3C Amaya. Les requêtes de type PUT sont beaucoup plus simples que les chargements de fichiers, et elles ressemblent à :

```
PUT /path/filename.html HTTP/1.1
```

Normalement, cela signifie que le client distant va sauver les données qui suivent dans le fichier : ``/path/filename.html'` de votre disque. Ce n'est évidemment pas très sécurisé de laisser Apache ou PHP écraser n'importe quel fichier de l'arborescence. Pour éviter ceci, il faut d'abord dire au serveur que vous voulez qu'un script PHP donné gère la requête. Avec Apache, il y a une directive pour cela : *Script*. Elle peut être placée n'importe où dans le fichier de configuration d'Apache. En général, les webmestres la place dans le bloc `<Directory>`, ou peut être dans le bloc `<Virtualhost>`. La ligne suivante fera très bien l'affaire :

```
Script PUT /put.php3
```

Elle indique à Apache qu'il doit envoyer les requêtes de chargement par méthode PUT au script `put.php3`. Bien entendu, cela présuppose que vous avez activé PHP pour qu'il prenne en charge les fichiers de type `.php3`, et que PHP est actif.

Dans le fichier `put.php3` file vous pouvez mettre ceci :

```
<?php
copy( $PHP_UPLOADED_FILE_NAME, $DOCUMENT_ROOT.$REQUEST_URI );
?>
```

Ce script va copier le fichier chargé par le client distant à l'endroit désiré. Vous aurez probablement à effectuer quelques tests et des authentifications d'utilisateur, avant d'effectuer cette copie. Le seul piège est que lorsque PHP reçoit un chargement par méthode PUT, il va enregistrer le fichier dans le dossier temporaire, tout comme avec la [8.4.1 Chargements de fichiers par méthode POST](#). A la fin de la requête, le fichier sera effacé. Ce qui fait que ce script doit placer le fichier chargé quelque part. Le nom du fichier temporaire est placé dans la variable globale `$PHP_PUT_FILENAME`, et la destination prévue est placée dans `$REQUEST_URI` (ces noms peuvent changer d'une configuration d'Apache à l'autre). Cette destination est celle qui est demandée par le client, et vous n'avez pas à obéir aveuglément au client. Vous pourriez par exemple, déplacer le fichier dans un dossier de chargement.

8.5 Authentification HTTP avec PHP

[\[Notes en ligne\]](#)

Les fonctions d'authentification *HTTP* de PHP ne sont disponibles que si PHP est exécuté comme module Apache, et non pas sous la forme d'un CGI. Sous cette forme, il est possible d'utiliser la fonction `header()` pour demander une authentification ("Authentication Required") au client, générant ainsi l'apparition d'une fenêtre de demande d'utilisateur et de mot de passe. Une fois que les champs ont été remplis, l'URL sera de nouveau appelée, avec les variables `$PHP_AUTH_USER`, `$PHP_AUTH_PW` et `$PHP_AUTH_TYPE` contenant respectivement le nom d'utilisateur, le mot de passe et le type d'authentification. Actuellement, seule l'authentification simple ("Basic") est supportée. Reportez vous à la fonction `header()` pour plus d'informations.

Voici un exemple de script qui force l'authentification du client pour accéder à une page :

Exemple d'authentification HTTP

```
<?php
if(!isset($PHP_AUTH_USER)) {
Header("WWW-Authenticate: Basic realm=\"My Realm\"");
Header("HTTP/1.0 401 Unauthorized");
echo "Texte à envoyer si le client appuie sur le bouton d'annulation\n";
exit;
} else {
echo "Bonjour $PHP_AUTH_USER.<P>"
echo "Vous avez entré le mot de passe $PHP_AUTH_PW.<P>"
}
?>
```

Au lieu d'afficher simplement les variables globales ***\$PHP_AUTH_USER*** et ***\$PHP_AUTH_PW***, vous préférerez sûrement vérifier la validité du nom d'utilisateur et du mot de passe. Par exemple, en envoyant ces informations à une base de données, ou en recherchant dans un fichier dbm.

Méfiez vous des navigateurs buggés, tels que Internet Explorer. Ils semblent très suceptibles concernant l'ordre des entêtes. Envoyer l'entête d'authentification (*WWW-Authenticate*) avant le code de HTTP / 1 . 0 401 semble lui convenir jusqu'à présent.

Pour éviter que quelqu'un écrive un script qui révèle les mots de passe d'une page, à la quelle on a accédé par une authentification traditionnelle, les variables globales ***PHP_AUTH*** ne seront pas assignées si l'authentification externe a été activée pour cette page. Dans ce cas, la variable ***\$REMOTE_USER*** peut être utilisée pour identifier l'utilisateur à l'extérieur.

Notez cependant que les manipulations ci-dessus n'empêchent pas quiconque qui possède une page non authentifiée de voler les mots de passes des pages protégées, sur le même serveur.

Netscape et Internet Explorer effaceront le cache d'authentification client si ils recoivent une réponse 401. Cela permet de déconnecter un utilisateur, pour le forcer à ré-entrer son nom de compte et son mot de passe. Certains programmeurs l'utilisent pour donner un délai d'expiration, ou alors, fournissent un bouton de déconnexion.

Authentification HTTP avec nom d'utilisateur/mot de passe forcé

```
<?php
function authenticate() {
Header( "WWW-authenticate: basic realm='Test Authentication System'");
Header( "HTTP/1.0 401 Unauthorized");
echo "Vous devez entrer un nom d'utilisateur valide et un mot de passe correct pour accéder à ce";
exit;
}
if(!isset($PHP_AUTH_USER) || ($SeenBefore == 1 38; !strcmp($OldAuth, $PHP_AUTH_USER)) ) {
authenticate();
}
else {
echo "Bienvenue $PHP_AUTH_USER<BR>";
echo "Old: $OldAuth";
echo "<FORM ACTION=\"\$PHP_SELF\" METHOD=POST>\n";
echo "<INPUT TYPE=HIDDEN NAME=\"SeenBefore\" VALUE=\"1\">\n";
echo "<INPUT TYPE=HIDDEN NAME=\"OldAuth\" VALUE=\"\$PHP_AUTH_USER\">\n";
echo "<INPUT TYPE=Submit VALUE=\"Re Authenticate\">\n";
echo "</FORM>\n";
}
?>
```

Ce comportement n'est pas nécessaire par le standard d'authentification HTTP Basic. Les tests avec Lynx ont montré qu'il n'affectait pas les informations de session lors de la réception d'un message de type 401, ce qui fait que passer ces informations entre le serveur et le client, et donnera l'accès à la ressource. Cependant, l'utilisateur peut utiliser la touche '_' pour détruire les anciennes autentications.

Notez aussi que tout ceci ne fonctionne pas sous Microsoft IIS et que les limitations de PHP en version CGI sont dues aux limitations de IIS.

8.6 Création d'images

[\[Notes en ligne\]](#)

PHP n'est pas limité à la création de fichier HTML. Il peut aussi servir à créer des images GIF, PNG, JPG, wbmp et xpm, à la volée, aussi bien pour les émettre que pour les sauver. Il faut alors compiler PHP avec la librairie GD. GD et PHP requièrent aussi d'autres librairies, suivant le format d'images que vous voulez supporter. GD a cessé de supporter le format GIF depuis la version 1.6.

Création d'images GIF avec PHP

```
<?php
header("Content-type: image/png");
$string=implode($argv, " ");
$im = imagecreatefrompng("images/button1.png");
$orange = imagecolorallocate($im, 220, 210, 60);
$px = (imagesx($im)-7.5*strlen($string))/2;
imagestring($im,3,$px,9,$string,$orange);
imagepng($im);
imagedestroy($im);
?>
```

Cet exemple sera appelé depuis une page HTML avec une balise telle que: ``. Le script ci-dessus récupère le texte de la chaîne \$string et l'ajoute sur l'image de fond "images/button1.gif". Le résultat est alors envoyé au client. C'est un moyen très pratique d'éviter d'avoir à redessiner des boutons à chaque fois que le texte du bouton change. Avec ce script, il est généré dynamiquement.

8.7 Connexions persistantes aux bases de données

[\[Notes en ligne\]](#)

Les connexions persistantes aux bases de données SQL sont des connexions qui ne se referment pas à la fin du script. Lorsqu'une connexion persistante est demandée, PHP s'assure qu'il n'y a pas une autre connexion identique (qui serait ouverte précédemment, avec le même nom d'hôte, d'utilisateur et le même mot de passe), et si une telle connexion existe, elle est utilisée. Sinon, elle est créée. Une connexion identique est une connexion qui a ouvert le même hôte, avec le même nom et même mot de passe (si ils sont nécessaires). Ceux qui ne sont pas rompus aux techniques des serveurs web et leur distribution de la charge de travail, se font parfois une fausse idée de ces connexions persistantes. En particulier, les connexions persistantes ne permettent pas l'ouverture de plusieurs sessions avec le même lien, ne permettent pas la réalisation de transactions efficaces et ne font pas le café. En fait, pour être extrêmement clair sur le sujet, les connexions persistantes ne vous donnent aucune fonctionnalité de plus que les connexions non persistantes.

Alors pourquoi?

Cela s'explique par la manière avec laquelle les serveurs web fonctionnent. Il y a trois manières d'utiliser PHP pour générer des pages.

La première est d'utiliser PHP comme un CGI (Common Interface Gateway). Lorsque PHP fonctionne de cette manière, une instance de l'interpréteur PHP est créée puis détruit pour chaque page demandée. Etant donné qu'il est détruit après chaque requête, toutes les ressources acquises (comme une connexion à une base SQL), sont purement et simplement détruites.

La deuxième méthode, et de loin, la plus prisée, est d'exécuter PHP sous la forme d'un module sur un serveur multi-process, ce qui revient à dire : Apache. Un tel serveur a typiquement un processus parent qui

coordonne un ensemble de processus fils, qui servent les fichiers. Lorsque les requêtes parviennent depuis un client, elles sont transmises à un fils disponible. Cela signifie que si un client fait une deuxième requête, il peut être servi par un processus client différent du premier. Les connexions persistantes vous permettent alors de ne vous connecter à une base SQL que la première fois. Lors des connexions ultérieures, les processus fils pourront réutiliser la connexion ouverte précédemment.

La dernière méthode est d'utiliser PHP sous la forme d'un module. Pour un serveur multi-thread, actuellement, c'est purement théorique, car PHP ne fonctionne par encore sous cette forme. Le développement est en cours pour supporter ISAPI, WSAPI, et NSAPI (sous Windows), qui permettront d'utiliser PHP comme un module pour des serveurs tels que Netscape FastTrack, Microsoft's Internet Information Server (IIS), et O'Reilly's WebSite Pro. Lorsque cela sera fait, le comportement sera le même que pour les serveur multi-process.

Si les connexions persistantes n'ont aucune fonctionnalité de plus, à quoi servent-elles?

La réponse est extrêmement simple : efficacité. Les connexions persistantes sont un bon moyen d'accélérer les accès à une base SQL si le traitement de connexion à la base est long. Ce temps dépend de nombreux facteurs : le type de base de données, cette base est-elle sur le même serveur ou pas, quelle est la charge du serveur de base de données, etc... Si le temps de connexion est long, les connexions persistantes seront bien utiles, car une fois ouverte par un processus fils, la connexion est réutilisable sans avoir à se reconnecter. Si vous avez 20 processus fils, il suffit d'avoir 20 connexions persistantes ouvertes, une par fils.

Notez que les connexions persistantes ont quelques inconvénients si vous hébergez une base de données, dont le nombre maximal de connexion risque d'être atteint par les connexions persistantes. Si votre base de données accepte jusqu'à 16 connexions simultanées et que, 17 processus essaient de se connecter, le dernier restera sur la touche. Si il y a des erreurs dans les scripts qui ne permettent pas de fermer la connexion (par exemple, une boucle infinie), votre serveur sera rapidement engorgé. Vérifier la documentation de votre base de données pour savoir comment elle traite les connexions inactives ou abandonnées.

Résumons nous : les connexions persistantes ont été définies pour avoir les mêmes fonctionnalités que les connexions non persistantes. Les deux types de connexions sont parfaitement interchangeables, et n'affecteront pas le comportement de votre script : uniquement son efficacité.

8.8 Utilisation des fichiers à distance

[\[Notes en ligne\]](#)

Aussi longtemps que le support de la fonction d'ouverture générique de fichiers ("URL fopen wrapper") est actif lorsque vous configurez PHP (il est inutile de passer explicitement l'option `--disable-url-fopen-wrapper` pour faire la configuration), vous pouvez utiliser des URLs (HTTP et FTP) avec la plupart des fonctions qui utilisent un nom de fichier comme paramètre, ceci incluant les expressions [require\(\)](#) et [include\(\)](#). Note : *Vous ne pouvez pas utiliser les fichiers distants dans les expressions [include\(\)](#) et [require\(\)](#) avec Windows.*

Par exemple, vous pouvez suivre l'exemple suivant pour ouvrir un fichier sur un serveur web distant, analyser les résultats pour extraire les informations dont vous avez besoin, et ensuite l'utiliser dans une requête de base de données, ou simplement éditer les informations dans le style de votre site.

Connaître le titre d'une page distante

```
<?php
$file = fopen("http://www.php.net/", "r");
if (!$file) {
echo "<p>Impossible d'ouvrir le fichier distant.\n";
exit;
}
while (!feof($file)) {
```

```

    $line = fgets($file, 1024);
    /* Cela ne fonctionne que si le titre est écrit sur une ligne.*/
    if (eregi("<title>(.)</title>", $line, $out)) {
        $title = $out[1];
        break;
    }
}
fclose($file);
?>

```

Vous pouvez aussi écrire des fichiers sur un serveur FTP aussi longtemps que vous êtes connecté avec un utilisateur ayant les bons droits d'accès, alors que le fichier n'existait pas encore. Pour vous connecter avec un utilisateur autre qu'anonyme, vous devez spécifier un nom d'utilisateur (et certainement le mot de passe) dans l'URL, comme par exemple 'ftp://user:password@ftp.example.com/path/to/file'. (Vous pouvez utiliser le même type de syntaxe pour accéder aux fichiers via **HTTP** lorsqu'ils nécessitent une authentification basique.)

Stocker des données sur un serveur distant

```

<?php
    $file = fopen("ftp://ftp.php.net/incoming/outputfile", "w");
    if (!$file) {
        echo "<p>Impossible d'ouvrir un fichier distant en écriture.\n";
        exit;
    }
    /* Ecriture des données. */
    fputs($file, "$HTTP_USER_AGENT\n");
    fclose($file);
?>

```

Note : Remarque: Vous pouvez avoir l'idée, à partir de l'exemple ci-dessus, d'utiliser la même technique pour écrire sur un log distant, mais comme mentionné ci-dessus vous ne pouvez qu'écrire sur un nouveau fichier en utilisant les fonctions [fopen\(\)](#) avec une URL. Pour faire des log distribués, nous vous conseillons de regarder la partie [syslog\(\)](#).

9 Langage

[\[Notes en ligne\]](#)

9.1 La syntaxe de base

[\[Notes en ligne\]](#)

9.1.1 Le passage du HTML au PHP

[\[Notes en ligne\]](#)

Il y a quatre moyens pour passer du mode HTML au mode PHP :

Le passage du HTML au PHP

1. `<? echo ("Ceci est un exemple d'affichage à l'écran en PHP, sous forme d'expression SGML.\n")`
2. `<?php echo("Si vous voulez afficher du XML ou du XHTML, faites comme ceci.\n"); ?>`
3. `<script language="php">`
`echo ("Certains éditeur HTML (comme FrontPage)`
`n'accepte pas les expressions telles que celle ci.");`
`</script>`
4. `<% echo ("Vous pouvez aussi utiliser le style ASP comme délimiteur."); %>`
`<%= $variable; # ceci est un raccourci pour "<%%echo .." %>`

La deuxième méthode est généralement utilisée, car elle permet une implémentation aisée de PHP avec la prochaine génération de XHTML.

La première possibilité n'est valable que si vous l'avez activée. Soit en faisant appel à la fonction `short_tags()` (NdT : semble avoir disparu), soit en utilisant l'option d'exécution [7.1.1.24 ini.short-open-tag](#) dans le fichier de configuration, soit en utilisant l'option de compilation `--enable-short-tags`.

La quatrième possibilité est seulement disponible si vous l'avez activée en utilisant soit l'option d'exécution [7.1.1.2 ini.asp-tags](#), soit en utilisant l'option de compilation `--enable-asp-tags`. Note : *Le support de la quatrième possibilité, ASP-style, a été ajouté dans la version 3.0.4.*

La marque de fermeture d'un bloc (`?>`) comprend la nouvelle ligne suivante, s'il y en a une.

9.1.2 Le séparateur d'instruction

[\[Notes en ligne\]](#)

Les instructions sont séparées comme en C ou en Perl par un point virgule à chaque fin d'instruction.

La balise de fin (`?>`) implique la fin d'un instruction, et donc ajoute implicitement un point virgule. Les deux exemples suivants sont équivalents.

```
<?php
echo "Ceci est un test";
?>
<?php echo "Ceci est un test" ?>
```

9.1.3 Commentaires

[\[Notes en ligne\]](#)

Le PHP supporte les commentaires comme en C, C++ et Shell Unix. Par exemple:

```
<?php
echo "Ceci est un test"; // Ceci est un commentaire sur une ligne comme en C++
    /* Ceci est un commentaire sur plusieurs lignes,
comme en C et C++ */
echo "Ceci est encore un test";
echo "Enfin, le test final"; # Ceci est un commentaire comme en Shell Unix
?>
```

Le premier type de commentaire ne commente que jusqu'à la fin de la ligne ou bien jusqu'à la fin du bloc, cela dépend du premier rencontré.

```
<h1>Ceci est un <?php echo "simple";?> exemple.</h1>
<p>La ligne du dessus affichera 'Ceci est un exemple'.
```

Faites attention à ne pas emboîter les commentaires de type 'C', ce qui arrive de temps en temps lorsque vous voulez commenter une grande partie de code.

```
<?php
/*
echo "Ceci est un test"; /* Ce commentaire va poser un problème */
*/
?>
```

9.2 Les constantes

[\[Notes en ligne\]](#)

PHP définit un certain nombre de constantes et propose des mécanismes pour en définir d'autres durant l'exécution. Les constantes se comportent des variables, à l'exception du fait que leur valeur est définie grâce à la fonction [define\(\)](#), et qu'elle ne peut pas être modifiée par la suite.

Les constantes prédéfinies (toujours disponibles) sont :

`__FILE__`

- Le nom du fichier qui est actuellement exécuté. Si cette constante est utilisée dans le cadre d'un fichier "inclus" (après utilisation de [require\(\)](#)), alors le nom du fichier inclus est renvoyé, et non le nom du fichier parent.

`__LINE__`

- Le numéro de la ligne qui est actuellement exécutée. Si cette constante est utilisée dans le cadre d'un fichier "inclus" (après utilisation de [require\(\)](#)), c'est la position dans le fichier inclus qui est renvoyé.

`PHP_VERSION`

- La chaîne de caractères de présentation de la version du PHP qui est actuellement utilisée. Par exemple '4.0.0'.

`PHP_OS`

- Nom du système d'exploitation qui est utilisé par la machine qui fait tourner le PHP. Par exemple, 'Linux'.

`TRUE`

- La valeur TRUE.
FALSE
- La valeur FALSE.
E_ERROR
- Dénote une erreur autre qu'une "parsing error" (erreur d'analyse) qu'il n'est pas possible de corriger.
E_WARNING
- Dénote un contexte dans lequel le PHP trouve que quelque chose qui ne va pas. Mais l'exécution se poursuit tout de même. Ces alertes-là peuvent être récupérées par le script lui-même. Un exemple serait une expression régulière (regexp) invalide dans la fonction [ereg\(\)](#).
E_PARSE
- L'analyseur a rencontré une forme syntaxique invalide dans le script. Correction de l'erreur impossible.
E_NOTICE
- Quelque chose s'est produit, qui peut être ou non une erreur. L'exécution continue. Par exemple, le cas de guillemets doubles (") non refermés, ou bien la tentative d'accéder à une variable qui n'est pas encore affectée.
E_ALL
- Toutes les constantes E_* rassemblées en une seule. Si vous l'utilisez avec [error_reporting\(\)](#), toutes les erreurs et les problèmes que PHP rencontrera seront notifiés.

Les constantes E_* sont généralement utilisées avec la fonction [error_reporting\(\)](#).

Vous pouvez définir d'autres constantes en utilisant la fonction [define\(\)](#).

Il est à noter que ce sont des constantes, et non pas des macros comme en C. Seulement les données scalaires peuvent être représentées par des constantes.

Définition de constantes

```
<?php
define("CONSTANT", "Bonjour le monde.");
echo CONSTANT; // affiche "Bonjourle monde."
?>
```

Utilisation des constantes __FILE__ et __LINE__

```
<?php
function report_error($file, $line, $message) {
    echo "Une erreur a eu lieu dans le fichier $file à la ligne $line: $message.";
}
report_error(__FILE__, __LINE__, "Y a un problème!");
?>
```

9.3 Les structures de contrôle

[\[Notes en ligne\]](#)

Tous les scripts PHP sont une suite d'instructions. Une instruction peut être une assignation, un appel de fonction, une instruction conditionnelle ou bien une instruction qui ne fait rien (une instruction vide). Une instruction se termine habituellement par un point virgule (;). De plus, plusieurs instructions peuvent être regroupées en bloc, délimité par des accolades ("{}"). Un bloc est considéré comme une instruction. Les différents types d'instruction sont décrits dans ce chapitre.

9.3.1 if

[\[Notes en ligne\]](#)

L'instruction `if` est une des plus importantes instructions de tous les langages, PHP inclus. Elle permet l'exécution conditionnelle d'une partie de code. Les fonctionnalités de l'instruction `if` sont les mêmes en PHP qu'en C :

```
<?php
if (expression)
commandes
?>
```

Comme nous l'avons vu dans le paragraphe consacré aux expressions, *expr* est évaluée à sa vraie valeur. Si l'expression *expr* est TRUE, PHP exécutera l'instruction et si elle est FALSE, l'instruction sera ignorée. L'exemple suivant affiche la phrase `a est plus grand que b` si *\$a* est plus grand que *\$b*:

```
<?php
if ($a > $b)
print "a est plus grand que b";
?>
```

Souvent, vous voulez que plusieurs instructions soient exécutées après un branchement conditionnel. Bien évidemment, il n'est pas obligatoire de répéter l'instruction conditionnelle autant de fois que vous avez d'instructions à exécuter. A la place, vous pouvez rassembler toutes les instructions dans un bloc. L'exemple suivant affiche `a est plus grand que b`, et assigne la valeur de la variable *\$a* à la variable *\$b*:

```
<?php
if ($a > $b) {
print "a est plus grand que b";
    $b = $a;
}
?>
```

Vous pouvez imbriquer indéfiniment des instructions `if` les unes dans les autres, ce qui permet une grande flexibilité dans l'exécution d'une partie de code suivant un grand nombre de conditions.

9.3.2 else

[\[Notes en ligne\]](#)

Souvent, vous voulez exécuter une instruction si une condition est remplie, et une autre instruction si cette condition n'est pas remplie. C'est à cela que sert `else`. `else` fonctionne avec après un `if` et exécute les instructions correspondantes au cas où l'expression du `if` est FALSE. Dans l'exemple suivant, ce bout de code affiche `a est plus grand que b` si la variable *\$a* est plus grande que la variable *\$a*, et `a est plus petit que b` sinon:

```
<?php
if ($a > $b) {
print "a est plus grand que b";
} else {
print "a est plus petit que b";
}
```

```
}
?>
```

Les instructions après le `else` ne sont exécutées que si l'expression du `if` est `FALSE`, et si elle n'est pas suivie par l'expression `elseif`.

9.3.3 elseif

[\[Notes en ligne\]](#)

`elseif`, comme son nom l'indique, est une combinaison de `if` et `else`. Comme l'expression `else`, il permet d'exécuter une instruction après un `if` dans le cas où le "premier" `if` est évalué comme `FALSE`. Mais, à la différence de l'expression `else`, il n'exécutera l'instruction que si l'expression conditionnelle `elseif` est évaluée comme `TRUE`. L'exemple suivant affichera `a est plus grand que b`, `a est égal à b` ou `a est plus petit que b`:

```
<?php
if ($a > $b) {
    print "a est plus grand que b";
} elseif ($a == $b) {
    print "a est égal à b";
} else {
    print "a est plus petit que b";
}
?>
```

Vous pouvez avoir plusieurs `elseif` qui s'imbriquent les uns dans les autres, après un `if` initial. Le premier `elseif` qui sera évalué à `TRUE` sera exécuté. En PHP, vous pouvez aussi écrire "else if" en deux mots et son comportement sera identique à la version en un seul mot.

L'expression `elseif` est exécutée seulement si le `if` précédent et tout autre `elseif` précédent est évalués comme `FALSE`, et que votre `elseif` est évalué à `TRUE`.

9.3.4 Syntaxe alternative

[\[Notes en ligne\]](#)

Le PHP propose une autre manière de rassembler des instructions à l'intérieur d'un bloc, pour les fonctions de contrôle `if`, `while`, `for`, et `switch`. Dans chaque cas, le principe est de remplacer l'accolade d'ouverture par deux points (`:`) et l'accolade de fermeture par, respectivement, `endif;`, `endwhile;`, `endfor;`, ou `endswitch;`.

```
<?php if ($a == 5): ?>
A vaut 5
<?php endif; ?>
```

Dans l'exemple ci-dessus, le block HTML "A = 5" est inclus à l'intérieur d'un `if` en utilisant cette nouvelle syntaxe. Ce code HTML ne sera affiché que si la variable `$a` est égale à 5.

Cette autre syntaxe fonctionne aussi avec le `else` et `elseif`. L'exemple suivant montre une structure avec un `if`, un `elseif` et un `else` utilisant cette autre syntaxe:

```
<?php
if ($a == 5):
    print "a égale 5";
print "...";
```

```
elseif ($a == 6):
print "a égale 6";
print "!!!";
else:
print "a ne vaut ni 5 ni 6";
endif;
?>
```

Allez voir [9.3.5 while](#), [9.3.7 for](#), et [9.3.1 if](#) pour d'autres exemples.

9.3.5 while

[\[Notes en ligne\]](#)

La boucle while est le moyen le plus simple d'implémenter une boucle en PHP. Cette boucle se comporte de la même manière qu'en C. L'exemple le plus simple d'une boucle while est le suivant :

```
<?php
while (expression) commandes
?>
```

La signification d'une boucle while est très simple. Le PHP exécute l'instruction tant que l'expression de la boucle while est évaluée comme TRUE. La valeur de l'expression est vérifiée à chaque début de boucle, et, si la valeur change durant l'exécution de l'instruction, l'exécution ne s'arrêtera qu'à la fin de l'itération (chaque fois que le PHP exécute l'instruction, on appelle cela une itération). De temps en temps, si l'expression du while est FALSE avant la première itération, l'instruction ne sera jamais exécutée.

Comme avec le if, vous pouvez regrouper plusieurs instructions dans la même boucle while en les regroupant à l'intérieur de parenthèses ou en utilisant la syntaxe suivante:

```
<?php
while (expression): commandes ... endwhile;
?>
```

Les exemples suivants sont identiques, et affichent tous les nombres de 1 à 10:

```
<?php
/* exemple 1 */
$i = 1;
while ($i <= 10) {
print $i++; /* La valeur affiche est $i avant l'incréméntation
              (post-incréméntation) */
}
/* exemple 2 */
$i = 1;
while ($i <= 10):
print $i;
    $i++;
endwhile;
?>
```


9.3.6 do..while

[\[Notes en ligne\]](#)

Les boucles do..while ressemblent beaucoup aux boucles while, mais l'expression est testée à la fin de chaque itération plutôt qu'au début. La principale différence par rapport à la boucle while est que la première itération de la boucle do..while est toujours exécutée (l'expression n'est testée qu'à la fin de l'itération), ce qui n'est pas le cas lorsque vous utilisez une boucle while (l'expression est vérifiée au début de chaque itération).

Il n'y a qu'une syntaxe possible pour les boucles do..while:

```
<?php
$i = 0;
do {
    print $i;
} while ($i>0);
?>
```

La boucle ci-dessus ne va être exécutée qu'une seule fois, car lorsque l'expression est évaluée, elle vaut FALSE (car la variable \$i n'est pas plus grande que 0) et l'exécution de la boucle s'arrête.

Les utilisateurs familiers du C sont habitués à une utilisation différente des boucles do..while, qui permet de stopper l'exécution de la boucle au milieu des instructions, en l'encapsulant dans un do..while(0) la fonction [9.3.9 break](#). Le code suivant montre une utilisation possible:

```
<?php
do {
    if ($i < 5) {
        print "i n'est pas suffisamment grand";
        break;
    }
    $i *= $factor;
    if ($i < $minimum_limit) {
        break;
    }
    print "i est bon";
    ...process i...
} while(0);
?>
```

Ne vous inquiétez pas si vous ne comprenez pas tout correctement. Vous pouvez écrire des scripts très très puissants sans utiliser cette fonctionnalité.

9.3.7 for

[\[Notes en ligne\]](#)

Les boucles for sont les boucles les plus complexes en PHP. Elles fonctionnent comme les boucles for du langage C. La syntaxe des boucles for est la suivante:

```
<?php
for (expr1; expr2; expr3) statement
?>
```

La première expression (*expr1*) est évaluée (exécutée), quoi qu'il arrive au début de la boucle.

Au début de chaque itération, l'expression *expr2* est évaluée. Si l'évaluation vaut TRUE, la boucle continue et l'instruction est exécutée. Si l'évaluation vaut FALSE, l'exécution de la boucle s'arrête.

A la fin de chaque itération, l'expression *expr3* est évaluée (exécutée).

Les expressions peuvent éventuellement être laissées vides. Si l'expression *expr2* est laissée vide, cela signifie que c'est une boucle infinie (PHP considère implicitement qu'elle vaut TRUE, comme en C). Cela n'est pas vraiment très utile, à moins que vous souhaitiez terminer votre boucle par l'instruction conditionnelle [9.3.9 break](#).

Considérons les exemples suivants. Tous affichent les chiffres de 1 à 10:

```
<?php
/* exemple 1 */
for ($i = 1; $i <= 10; $i++) {
    print $i;
}
/* exemple 2 */
for ($i = 1;;$i++) {
    if ($i > 10) {
        break;
    }
    print $i;
}
/* exemple 3 */
$i = 1;
for (;;) {
    if ($i > 10) {
        break;
    }
    print $i;
    $i++;
}
/* exemple 4 */
for ($i = 1; $i <= 10; print $i, $i++);
?>
```

Bien évidemment, le premier exemple est le plus simple de tous (ou peut être le quatrième), mais vous pouvez aussi penser qu'utiliser une expression vide dans une boucle for peut être utile parfois.

PHP supporte aussi la syntaxe alternative suivante pour les boucles for :

```
<?php
for (expr1; expr2; expr3): statement; ...; endfor;
?>
```

Les autres langages ont l'instruction foreach pour accéder aux éléments d'un tableau. PHP 3 ne dispose pas d'une telle fonction; PHP 4 en dispose (voir [9.3.8 foreach](#)). En PHP 3, vous pouvez combiner [9.3.5 while](#) avec [list\(\)](#) et [each\(\)](#) pour obtenir le même résultat. Reportez vous aux exemples de la documentation.

9.3.8 foreach

[\[Notes en ligne\]](#)

PHP 4 (mais pas PHP 3) inclut une commande foreach, comme en Perl ou d'autres langages. C'est un moyen simple de passer en revue un tableau. Il y a deux syntaxes possibles : la seconde est une extension mineure mais pratique de la première:

```
<?php
foreach(array_expression as $value) commandes
foreach(array_expression as $key => $value) commandes
?>
```

La première forme passe en revue le tableau `array_expression`. A chaque itération, la valeur de l'élément courant est assigné à `$value` et le pointeur interne de tableau est avancé d'un élément (ce qui fait qu'à la prochaine itération, on accédera à l'élément suivant).

La deuxième forme fait exactement la même chose, mais c'est la clé de l'élément courant qui est assigné à la variable `$key`.

Lorsque `foreach` démarre, le pointeur interne de fichier est automatiquement ramené au premier élément du tableau. Cela signifie que vous n'aurez pas à faire appel à [reset\(\)](#) avant `foreach`.

Vous pouvez remarquer que les exemples suivants fonctionnent de manière identique :

```
reset ($arr);
while (list(, $value) = each ($arr)) {
    echo "Valeur: $value<br>\n";
}
foreach ($arr as $value) {
    echo "Valeur: $value<br>\n";
}
```

Les exemples suivants sont aussi fonctionnellement identiques :

```
reset ($arr);
while (list($key, $value) = each ($arr)) {
    echo "Clé: $key; Valeur: $value<br>\n";
}
foreach ($arr as $key => $value) {
    echo "Clé: $key; Valeur: $value<br>\n";
}
```

Voici quelques exemples de plus :

```
<?php
/* exemple 1: valeur seule */
$a = array (1, 2, 3, 17);
foreach ($a as $v) {
    print "Valeur courante de \$a: $v.\n";
}
/* exemple 1: valeur (avec clé associée) */
$a = array (1, 2, 3, 17);
$i = 0; /* pour affichage seulement*/
foreach($a as $v) {
    print "\$a[$i] => $k.\n";
}
/* exemple 1: valeur et clé */
$a = array (
    "un" => 1,
    "deux" => 2,
    "trois" => 3,
    "dix-sept" => 17
);
foreach($a as $k => $v) {
    print "\$a[$k] => $v.\n";
}
```

?>

9.3.9 break

[\[Notes en ligne\]](#)

L'instruction break permet de sortir d'une structure if, for, while, ou switch.

break accepte un argument numérique optionnel qui vous indiquera combien de structures emboîtées ont été interrompues.

```
<?php
$i = 0;
while ($i < 10) {
    if ($arr[$i] == "stop") {
        break; /* Vous pouvez aussi écrire 'break 1;' ici. */
    }
    $i++;
}
/* Utilisation de l'argument optionnel. */
$i = 0;
while ( ++$i ) {
    switch ( $i ) {
        case 5:
            echo "à 5<br>\n";
            break 1; /* Ne sort que du switch. */
        case 10:
            echo "à 10; quitting<br>\n";
            break 2; /* Sort du switch et du while. */
        default:
            break;
    }
}
?>
```

9.3.10 continue

[\[Notes en ligne\]](#)

L'instruction continue est utilisée dans une boucle afin d'éluder les instructions de l'itération courante afin de passer directement à l'itération suivante.

continue accepte un argument numérique optionnel qui vous indiquera combien de structures emboîtées ont été ignorées.

```
while (list ($cle, $valeur) = each ($arr)) {
    if (!( $cle % 2)) { // évite les membres impairs
        continue;
    }
    fonction_quelconque($valeur);
}
$i = 0;
while ($i++ < 5) {
    echo "Dehors<br>\n";
    while (1) {
        echo " Milieu<br>\n";
```

```

while (1) {
echo "  Intérieur<br>\n";
continue 3;
}
echo "Ceci n'est jamais atteint.<br>\n";
}
echo "Ceci non plus.<br>\n";
}

```

9.3.11 switch

[\[Notes en ligne\]](#)

L'instruction switch équivaut à une série d'instructions if. En de nombreuses occasions, vous aurez besoin de comparer la même variable (ou expression) avec un grand nombre de valeurs différentes, et d'exécuter différentes parties de code suivant la valeur à laquelle elle est égale. C'est exactement à cela que sert l'instruction switch.

Les deux exemples suivants sont deux manières différentes d'écrire la même chose, l'une en utilisant une série de if, et l'autre en utilisant l'instruction switch:

```

if ($i == 0) {
print "i égale 0";
}
if ($i == 1) {
print "i égale 1";
}
if ($i == 2) {
print "i égale 2";
}
switch ($i) {
case 0:
print "i égale 0";
break;
case 1:
print "i égale 1";
break;
case 2:
print "i égale 2";
break;
}

```

Il est important de comprendre que l'instruction switch exécute chacune des clauses dans l'ordre. L'instruction switch est exécutée ligne par ligne. Au début, aucun code n'est exécuté. Seulement lorsqu'un case est vérifié, PHP exécute alors les instructions correspondantes. PHP continue d'exécuter les instructions jusqu'à la fin du bloc d'instructions du switch, ou bien dès qu'il trouve l'instruction break. Si vous ne pouvez pas utiliser l'instruction break à la fin de l'instruction case, PHP continuera à exécuter toutes les instructions qui suivent. Par exemple :

```

switch ($i) {
case 0:
print "i égale 0";
case 1:
print "i égale 1";
case 2:
print "i égale 2";
}

```

```
}
```

Dans cet exemple, si \$i est égal à 0, PHP va exécuter quand même toutes les instructions qui suivent. Si \$i est égal à 1, PHP exécutera les deux dernières instructions. Et seulement si \$i est égal à, vous obtiendrez le résultat escompté, c'est-à-dire, l'affiche de "i égal 2. Donc, l'important est de ne pas oublier l'instruction break (même si il est possible que vous l'omettiez dans certaines circonstances).

Dans une commande switch, une condition n'est évaluée qu'une fois, est le résultat est comparé à chaque case. Dans une structure elseif, les conditions sont évaluées à chaque comparaison. Si votre condition est plus compliquée qu'une simple comparaison, ou bien fait partie d'une boucle, switch sera plus rapide.

La liste de commande d'un case peut être vide, auquel cas PHP utilisera la liste de commandes du cas suivant.

```
switch ($i) {
case 0:
case 1:
case 2:
print "i est plus petit que 3 mais n'est pas négatif";
break;
case 3:
print "i égale 3";
}
```

Un case spécial est default. Ce cas est utilisé lorsque tous les case ont échoués. Par exemple :

```
switch ($i) {
case 0:
print "i égale 0";
break;
case 1:
print "i égale 1";
break;
case 2:
print "i égale 2";
break;
default:
print "i n'est ni égal à 2, ni à 1, ni à 0.";
}
```

Une autre chose à mentionner est que l'instruction case peut être une expression à de type scalaire, c'est-à-dire nombre entier, nombre à virgule flottante et chaîne de caractère. Les tableaux sont sans intérêt dans ce contexte-là.

La syntaxe alternative pour cette structure de contrôle est la suivante : pour plus d'informations, voir [9.3.4 Syntaxe alternative](#)).

```
switch ($i):
case 0:
print "i égale 0";
break;
case 1:
print "i égale 1";
break;
case 2:
print "i égale 2";
break;
default:
```

```
print "i n'est ni égal à 2, ni à 1, ni à 0";
endswitch;
```

9.3.12 [require\(\)](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

La commande [require\(\)](#) se remplace elle-même par le contenu du fichier spécifié, comme les préprocesseurs C le font avec la commande `#include`.

Il est important de noter que lorsqu'un fichier est [include\(\)](#) ou [require\(\)](#), les erreurs d'analyse apparaîtront en HTML tout au début du fichier, et l'analyse du fichier parent ne sera pas interrompue. Pour cette raison, le code qui est dans le fichier doit être placé entre [9.1.1 Le passage du HTML au PHP](#).

[require\(\)](#) n'est pas vraiment une fonction PHP : c'est plus une instruction du langage. Elle ne fonctionne pas comme les fonctions standards. Par exemple, [require\(\)](#) ne peut pas contenir d'autres structures de contrôle. De plus, il ne retourne aucune valeur. Lire une valeur retournée par un [require\(\)](#) retourne une erreur d'analyse. Contrairement à [include\(\)](#), [require\(\)](#) *va toujours* lire dans le fichier cible, même si la ligne n'est jamais exécutée. Si vous souhaitez une inclusion conditionnelle, utilisez [include\(\)](#). La condition ne va jamais affecter [require\(\)](#). Cependant, si la ligne de [require\(\)](#) n'est jamais exécutée, le code du fichier ne le sera jamais non plus.

Les boucles n'affectent pas le comportement de [require\(\)](#). Même si le code contenu dans le fichier source est appelé dans la boucle, [require\(\)](#) n'est exécuté qu'une fois.

Cela signifie qu'on ne peut pas mettre un [require\(\)](#) dans une boucle, et s'attendre à ce qu'il inclue du code à chaque itération. Pour cela, il faut utiliser [include\(\)](#).

```
require ('header.inc');
```

Attention : [include\(\)](#) et [require\(\)](#) ajoute le contenu du fichier cible dans le script lui-même. Elle n'utilise pas le protocole HTTP ou tout autre protocole. Toute variable qui est dans le champs du script sera accessible dans le fichier d'inclusion, et vice versa.

```
require ("file.inc?varone=1&vartwo=2"); /* Ne fonctionne pas. */
$varone = 1;
$vartwo = 2;
require ("file.inc"); /* $varone et $vartwo seront accessibles à file.inc */
```

Ne vous laissez pas abuser par le fait que vous pouvez requérir ou inclure des fichiers via HTTP en utilisant la fonctionnalité de [8.8 Utilisation des fichiers à distance](#) ce qui est au dessus reste vrai.

En PHP 3, il est possible d'exécuter une commande `return` depuis un fichier inclus, tant que cette commande intervient au niveau global du fichier inclus. Elle ne doit intervenir dans aucun bloc (entre accolade `{ }`). En PHP 4, cette possibilité a été supprimée. Si vous en avez besoin, utilisez plutôt [include\(\)](#).

9.3.13 [include\(\)](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

La fonction [include\(\)](#) inclus et évalue le fichier spécifié en argument.

Il est important de noter que lorsqu'un fichier est [include\(\)](#) ou [require\(\)](#), les erreurs d'analyse apparaîtront en HTML tout au début du fichier, et l'analyse du fichier parent ne sera pas interrompue. Pour cette raison, le code qui est dans le fichier doit être placé entre [9.1.1 Le passage du HTML au PHP](#).

Cela a lieu à chaque fois que la fonction [include\(\)](#) est rencontrée. Donc vous pouvez utiliser la fonction [include\(\)](#) dans une boucle pour inclure un nombre infini de fois un fichier, ou même des fichiers différents.

```
<?php
$files = array ('premier.inc', 'second.inc', 'troisieme.inc');
for ($i = 0; $i < count($files); $i++) {
include $files[$i];
}
?>
```

[include\(\)](#) diffère de [require\(\)](#) car le fichier inclus est ré-évalué à chaque fois que la commande est exécutée, tandis que [require\(\)](#) est remplacée par le fichier cible lors de la première exécution, que son contenu soit utilisé ou non. De plus, cela se fait même si il est placé dans une structure conditionnelle, comme dans un [9.3.1 if](#)).

Parce que la fonction [include\(\)](#) nécessite une construction particulière, vous devez l'inclure dans un bloc si elle est incluse dans une structure conditionnelle.

```
<?php
/* Ceci est faux, et ne fonctionnera pas ce qu'on attends. */
if ($condition)
include($file);
else
include($other);
/* Ceci est CORRECT. */
if ($condition) {
include($file);
} else {
include($other);
}
?>
```

En PHP 3, il est possible d'exécuter une commande return depuis un fichier inclus, tant que cette commande intervient au niveau global du fichier inclus. Elle ne doit intervenir dans aucun bloc (entre accolade {}). En PHP 4, cette possibilité a été supprimée. Cependant, PHP 4 vous autorise à retourner des valeurs d'un fichier inclus. Vous pouvez traiter [include\(\)](#) comme une fonction normale, qui retourne une valeur. Mais cela génère une erreur d'analyse en PHP 3.

include() en php 3 et php 4

On suppose que le fichier `test.inc` existe, et est placé dans le même dossier que le fichier principal :

```
<?php
echo "Avant le retour<br>\n";
if (1) {
return 27;
}
echo "Après le retour <br>\n";
?>
```

On suppose que le fichier `main.html` contient ceci :


```
<?php
$retval = include ('test.inc');
echo "Fichier inclus: '$retval'<br>\n";
?>
```

Lorsque ``main.html`` est appelé en PHP 3, il va générer une erreur d'analyse (parse error) à la ligne 2; vous ne pouvez pas vous attendre à un retour sur une fonction [include\(\)](#) en PHP 3. En PHP 4, cependant, le résultat sera :

```
Avant le retour
Fichier inclus : '27'
```

Supposons maintenant que ``main.html`` a été modifié et contient maintenant le code suivant :

```
<?php
include ('test.inc');
echo "Retour dans le main.html<br>\n";
?>
```

En PHP 4, l'affichage sera :

```
Avant le retour
Retour dans le main.html
```

Au contraire, PHP 3 affichera :

```
Avant le retour
27Retour dans le main.html
Parse error: parse error in /home/torben/public_html/phptest/main.html on line 5
```

L'erreur d'analyse ci-dessus est le résultat du fait que la commande `return` est dans un bloc qui n'est pas une fonction, dans ``test.inc``. Lors que le `return` est sorti du bloc, l'affichage devient :

```
Avant le retour
27Retour dans le main.html
```

Le '27' est dû au fait que PHP 3 ne supporte pas le `return` dans ces fichiers.

Il est important de noter que lorsqu'un fichier est [include\(\)](#) ou [require\(\)](#), les erreurs d'analyse apparaîtront en HTML tout au début du fichier, et l'analyse du fichier parent ne sera pas interrompue. Pour cette raison, le code qui est dans le fichier doit être placé entre [9.1.1 Le passage du HTML au PHP](#).

```
include ("file.inc?varone=1&vartwo=2"); /* ne fonctionne pas. */
$varone = 1;
$vartwo = 2;
include ("file.inc"); /* $varone et $vartwo sont accessibles dans file.inc */
```

Ne vous laissez pas abuser par le fait que vous pouvez requérir ou inclure des fichiers via HTTP en utilisant la fonctionnalité de [8.8 Utilisation des fichiers à distance](#) ce qui est au dessus reste vrai.

Voir aussi [readfile\(\)](#), [require\(\)](#) et [virtual\(\)](#).

9.3.14 require_once()

[\[Notes en ligne\]](#) [\[Exemples\]](#)

La commande [require_once\(\)](#) se remplace elle-même par le fichier spécifié, un peu comme les commandes de préprocesseur C `#include`, et ressemble sur ce point à [require\(\)](#). La principale différence est qu'avec [require_once\(\)](#), vous êtes assurés que ce code ne sera ajouté qu'une seule fois, évitant de ce fait les redéfinitions de variables ou de fonctions, génératrices d'alertes.

Par exemple, si vous créez les deux fichiers d'inclusion `utils.inc` et `foolib.inc`

utils.inc

```
<?php
define(PHPVERSION, floor(phpversion()));
echo "LES GLOBALES SONT SYMPAS\n";
function goodTea() {
return "Le Earl Grey est délicieux!";
}
?>
```

foolib.inc

```
<?php
require ("utils.inc");
function showVar($var) {
if (PHPVERSION == 4) {
print_r($var);
} else {
dump_var($var);
}
}
// Une série de fonctions
?>
```

Puis, vous écrivez un script `cause_error_require.php`

cause_error_require.php

```
<?php
require("foolib.inc");
/* Ceci génère une erreur*/
require("utils.inc");
$foo = array("1",array("complex","quaternion"));
echo "Ce code requiert utils.inc une deuxième fois, car il est requis \n";
echo "dans foolib.inc\n";
echo "Utilisation de GoodTea: ".goodTea()."\n";
echo "Affichage de foo: \n";
showVar($foo);
?>
```

Lorsque vous exécutez le script ci dessus, le résultat sera (sous PHP 4.01pl2):

```
GLOBALS ARE NICE
GLOBALS ARE NICE
Fatal error: Cannot redeclare causeerror() in utils.inc on line 5
```

En modifiant foolib.inc et cause_error_require.php pour qu'elles utilisent [require_once\(\)](#) au lieu de [require\(\)](#) et ne renommant le fichier en avoid_error_require_once.php, on obtient :

foolib.inc (corrigé)

```
<?php
require_once("utils.inc");
function showVar($var) {
?>
```

avoid_error_require_once.php

```
<?php
require_once("foolib.inc");
require_once("utils.inc");
$foo = array("1",array("complexe","quaternion"));
?>
```

L'exécution de ce script, sous PHP 4.0.1pl2, donne :

```
LES GLOBALES SONT SYMPA
Ce code requiert utils.inc une deuxième fois, car il est requis
dans foolib.inc
Utilisation de GoodTea: Le Earl Grey est délicieux!
Affichage de foo:
Array
(
    [0] => 1
    [1] => Array
        (
            [0] => complexe
            [1] => quaternion
        )
)
```

Notez aussi que, de la même manière que les préprocesseur traitent les `#include`, cette commande est exécutée au moment de la compilation, c'est à dire lorsque le script est analysée, et avant qu'il soit exécuté, et ne doit pas être utilisée pour insérer des données dynamiques liées à l'exécution. Il vaut alors mieux utiliser [include_once\(\)](#) ou [include\(\)](#).

Pour plus d'exemples avec [require_once\(\)](#) et [include_once\(\)](#), jetez un oeil dans le code de PEAR inclus dans la dernière distribution de PHP.

Voir aussi : [require\(\)](#), [include\(\)](#), [include_once\(\)](#), [get_required_files\(\)](#), [get_included_files\(\)](#), [readfile\(\)](#), et [virtual\(\)](#).

9.3.15 include_once()

[\[Notes en ligne\]](#) [\[Exemples\]](#)

La commande [include_once\(\)](#) inclus et évalue le fichier spécifié durant l'exécution du script. Le comportement est similaire à [include\(\)](#), mais la différence est que si le code a déjà été inclus, il ne le sera pas une seconde fois.

Comme précisé dans la section [require_once\(\)](#), la fonction [include_once\(\)](#) est utilisée de préférence lorsque le fichier doit être inclus ou évalué plusieurs fois dans un script, ou bien lorsque vous voulez être sur qu'il ne sera inclus qu'une seule fois, pour éviter des redéfinitions de fonction.

Pour plus d'exemples avec [require_once\(\)](#) et [include_once\(\)](#), jetez un oeil dans le code de PEAR inclus dans

la dernière distribution de PHP.

Voir aussi: [require\(\)](#), [include\(\)](#), [require_once\(\)](#), [get_required_files\(\)](#), [get_included_files\(\)](#), [readfile\(\)](#), et [virtual\(\)](#).

9.4 Les expressions

[\[Notes en ligne\]](#)

Les expressions sont la partie la plus importante du PHP. En PHP, presque tout ce que vous écrivez est une expression. La manière la plus simple de définir une expression est : "tout ce qui a une valeur".

Les formes les plus simples d'expressions sont les constantes et les variables. Lorsque vous écrivez "\$a = 5", vous assignez la valeur '5' à la variable \$a. Bien évidemment, '5' vaut 5 ou, en d'autres termes, '5' est une expression avec pour valeur 5 (dans ce cas, '5' est un entier constant).

Après cette assignation, vous pouvez considérer que \$a a pour valeur 5 et donc, écrire \$b = \$a, revient à écrire \$b = 5. En d'autres termes, \$a est une expression avec de valeur 5. Si tout fonctionne correctement, c'est exactement ce qui arrive.

Un exemple plus complexe concerne les fonctions. Par exemple, considérons la fonction suivante :

```
<?php
function foo () {
return 5;
}
?>
```

Considérant que vous êtes familier avec le concept de fonction, (si ce n'est pas le cas, jetez un oeil au chapitre concernant les fonctions), vous serez d'accord que \$c = foo() est équivalent à \$c = 5, et vous aurez tout à fait raison. Les fonctions sont des expressions qui ont la valeur de leur "valeur de retour". Si foo() renvoie 5, la valeur de l'expression 'foo()' est 5. Habituellement, les fonctions ne font pas que renvoyer une valeur constante mais réalisent des traitements.

Bien sur, les valeurs en PHP n'ont pas à être des valeurs numériques, comme c'est souvent le cas. PHP supporte 3 types de variables scalaires : les valeurs entières, les nombres à virgule flottante et les chaînes de caractères. (une variable scalaire est une variable que vous ne pouvez pas scinder en morceau, au contraire des tableaux par exemple). PHP supporte aussi deux types composés : les tableaux et les objets. Chacun de ces types de variables peuvent être affectés ou renvoyés par une fonction.

Les utilisateurs de PHP/FI 2 ne verront aucun changement. Malgré tout, PHP va plus loin dans la gestion des expressions, comme le font d'autres langages. PHP est un langage orienté expression, dans le sens où presque tout est une expression. Considérons l'exemple dont nous avons déjà parlé, '\$a = 5'. Il est facile de voir que il y a deux valeurs qui entrent en jeu ici, la valeur numérique constante '5' et la valeur de la variable \$a qui est mis à jour à la valeur 5. Mais, la vérité est qu'il y a une autre valeur qui entre en jeu ici et c'est la valeur de l'assignement elle-même. L'assignement lui-même est assigné à une valeur, dans ce cas-là 5. En pratique, cela signifie que '\$a = 5' est une expression qui a pour valeur 5. Donc, en écrire '\$b = (\$a = 5)' revient à écrire '\$a = 5; \$b = 5;' (un point virgule marque la fin d'une instruction). Comme les assignements sont analysés de droite à gauche, vous pouvez aussi bien écrire '\$b = \$a = 5'.

Un autre bon exemple du langage orienté expression est la pré-incrémentation et la post-incrémentation, (ainsi que la décrément). Les utilisateurs de PHP/FI 2 et ceux de nombreux autres langages sont habitués à la notation "variable++" et "variable--". Ce sont les opérateurs d'incrément et de décrément. En PHP/FI 2, l'instruction '\$a++' n'a aucune valeur (c'est-à-dire que ce n'est pas une expression) et vous ne pouvez donc pas l'utiliser. PHP ajoute les possibilités d'incrément et de décrément comme c'est le cas dans le langage C. En PHP, comme en C, il y a deux types d'opérateurs d'incrément (pré-incrément et post-incrément). Les deux types d'opérateur d'incrément jouent le même rôle (c'est-à-dire qu'il incrémente la variable). La différence vient de la valeur de l'opérateur

d'incréméntation. L'opérateur de pré-incréméntation, qui s'écrit '++\$variable', évalue la valeur incrémentée (PHP incréménte la variable avant de lire la valeur de cette variable, d'oú le nom de 'pré-incréméntation'). L'opérateur de post-incréméntation, qui s'écrit '\$variable++', évalue la valeur de la variable avant de l'incréménter. (PHP incréménte la variable après avoir lu sa valeur, d'oú le nom de 'post-incréméntation'). Un type d'expression très commun est l'expression de comparaison. Ces expressions sont évaluées à 0 ou 1, autrement dit FALSE ou TRUE (respectivement). PHP supporte les opérateurs de comparaison > (plus grand que), => (plus grand ou égal), == (égal à), < (plus petit que), <= (plus petit ou égal). Ces expressions sont utilisées de manière courante dans les instructions conditionnelles, comme l'instruction if.

Pour le dernier exemple d'expression, nous allons parler des combinaisons d'opérateurs/assignement. Vous savez que si vous voulez incréménter la variable \$a d'une unité, vous devez simplement écrire '\$a++'. Mais si vous voulez ajouter la valeur '3' à votre variable ? Vous pouvez écrire plusieurs fois '\$a++', mais ce n'est pas la meilleure des méthodes. Un pratique plus courante est d'écrire '\$a = \$a + 3'. L'expression '\$a + 3' correspond à la valeur \$a plus 3, et est de nouveau assignée à la variable \$a. Donc le résultat est l'incréméntation de 3 unités. En PHP, comme dans de nombreux autres langages comme le C, vous pouvez écrire cela de manière plus concise, manière qui avec le temps se révèlera plus claire et plus rapide à comprendre. Ajouter 3 à la valeur de la variable \$a peut s'écrire '\$a += 3'. Cela signifie précisément : "on prend la valeur de la variable \$a, on ajoute la valeur 3 et on assigne cette valeur à la variable \$a". Et pour être plus concis et plus clair, cette expression est plus rapide. La valeur de l'expression '\$a += 3', comme l'assignement d'une valeur quelconque, est la valeur assignée. Il est à noter que ce n'est pas 3 mais la combinaison de la valeur de la variable \$a plus la valeur 3. (c'est la valeur qui est assignée à la variable \$a). N'importe quel opérateur binaire peu utiliser ce type d'assignement, par exemple '\$a -= 5' (soustraction de 5 de la valeur de la variable \$a), '\$b *= 7' (multiplication de la valeur de la variable \$b par 7). Il y a une autre expression qui peut paraître complexe si vous ne l'avez pas vu dans d'autre langage, l'opérateur conditionnel ternaire:

```
<?php
$first ? $second : $third
?>
```

Si la valeur de la première sous-expression est vraie, (différente de 0), alors la deuxième sous-expression est évaluée et constitue le résultat de l'expression conditionnelle. Sinon, c'est la troisième sous-expression qui est évaluée et qui constitue le résultat de l'expression.

Les exemples suivants devraient vous permettre de mieux comprendre la pré- et post- incréméntation et le concept des expressions en général:

```
<?php
function double($i) {
    return $i*2;
}
$b = $a = 5;           /* assigne la valeur 5 aux variables $a et $b */
$c = $a++;             /* post-incréméntation de la variable $a et assignation de
la valeur à la variable $c */
$e = $d = ++$b;        /* Pré-incréméntation, et assignation de la valeur aux
variables $d et $e */
/* à ce niveau, les variables $d et $e sont égales à 6 */
$f = double($d++);     /* assignation du double de la valeur de $d à la variable $f ($f vaut 12),
puis incréméntation de la valeur de $d */
$g = double(++$e);     /* assigne deux fois la valeur de $e après
incréméntation, 2*7 = 14 to $g */
$h = $g += 10;         /* Tout d'abord, $g est incréméntée de 10, et donc $g vaut 24.
Ensuite, la valeur de $g, (24) est assignée à la variable $h,
qui vaut donc elle aussi 24. */
?>
```

Au début de ce chapitre, nous avons dit que nous allions décrire les différents types d'instructions, et donc, comme promis, nous allons voir que les expressions peuvent être des instructions. Mais, attention, toutes les expressions ne sont pas des instructions. Dans ce cas-là, une instruction est de la forme 'expr' ';', c'est-à-dire, une expression suivie par un point-virgule. L'expression '\$b = \$a = 5;', '\$a = 5' est valide, mais ce n'est pas une instruction en elle-même. '\$b = \$a = 5' est une instruction valide.

La dernière chose qui mérite d'être mentionnée est la véritable valeur des expressions. Lorsque vous faites des tests sur une variable, dans une boucle conditionnelle par exemple, cela ne vous intéresse pas de savoir qu'elle est la valeur exacte de l'expression. Mais vous voulez seulement savoir si le résultat signifie TRUE ou FALSE (PHP n'a pas de type booléen). La véritable valeur d'une expression en PHP est calculée de la même manière qu'en Perl. Toute valeur numérique différente de 0 est considérée comme étant TRUE. Une chaîne de caractères vide et la chaîne de caractère 0 sont considérées comme FALSE. Toutes les autres valeurs sont vraies. Avec les types de variables non-scalaires (les tableaux et les objets), s'ils ne contiennent aucun élément, renvoient FALSE, sinon, ils renvoient TRUE.

PHP propose une implémentation complète et détaillée des expressions. PHP documente toutes ses expressions dans le manuel que vous êtes en train de lire. Les exemples qui vont suivre devraient vous donner une bonne idée de ce qu'est une expression et comment construire vos propres expressions. Dans tout ce qui va suivre, nous écrirons *expr* pour indiquer toute expression PHP valide.

9.5 Fonctions

[\[Notes en ligne\]](#)

9.5.1 Les fonctions utilisateurs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

une fonction peut être définie en utilisant la syntaxe suivante :

```
<?php
function foo ($arg_1, $arg_2, ..., $arg_n) {
echo "Exemple de fonction.\n";
return $retval;
}
?>
```

Tout code PHP, correct syntaxiquement, peut apparaître dans une fonction et dans une définition de [9.6.1 Les classes : class](#).

En PHP 3, les fonctions doivent être définies avant qu'elles ne soient utilisées. Ce n'est plus le cas en PHP 4. PHP ne supporte pas le surchargement de fonction, ni la destruction ou la redéfinition de fonctions déjà déclarées.

PHP 3 ne supporte pas un nombre variable d'arguments (voir [9.5.2.2 Valeur par défaut des arguments](#) pour plus d'informations). PHP 4 supporte les deux : voir [9.5.2.3 Nombre d'arguments variable](#) et les fonctions de références que sont [func_num_args\(\)](#), [func_get_arg\(\)](#), et [func_get_args\(\)](#) pour plus d'informations.

9.5.2 Les arguments de fonction

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Des informations peuvent être passées à une fonction en utilisant un tableau d'arguments, dont chaque élément est séparé par une virgule. Un élément peut être une variable ou une constante.

PHP supporte le passage d'arguments [9.5.2.2 Valeur par défaut des arguments](#) (méthode par défaut), par [9.5.2.1 Passage d'arguments par référence](#). Les listes variable d'arguments sont supportées par PHP 4 et plus

récent. Voir [9.5.2.3 Nombre d'arguments variable](#) et les fonctions utiles que sont [func_num_args\(\)](#), [func_get_arg\(\)](#), et [func_get_args\(\)](#). Fonctionnellement, on peut arriver au même résultat en passant un tableau comme argument :

```
function takes_array($input) {
    echo "$input[0] + $input[1] = ", $input[0]+$input[1];
}
```

[9.5.2.1 Passage d'arguments par référence](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Par défaut, les arguments sont passés à la fonction par valeur (donc vous pouvez changer la valeur d'un argument dans la fonction, cela ne change pas sa valeur à l'extérieur de la fonction). Si vous voulez que vos fonctions puisse changer la valeur des arguments, vous devez passer ces arguments par référence. Si vous voulez qu'un argument soit toujours passé par référence, vous pouvez ajouter un '&' devant l'argument dans la déclaration de la fonction :

```
function add_some_extra(&$string) {
    $string .= ', et un peu plus.';
}
$str = 'Ceci est une chaîne';
add_some_extra($str);
echo $str;    // affiche 'Ceci est une chaîne, et un peu plus.'
```

Si vous souhaitez passer une variable par référence à une fonction mais de manière ponctuelle, vous pouvez ajouter un '&' devant l'argument dans l'appel de la fonction:

```
function foo ($bar) {
    $bar .= ', et un peu plus.';
}
$str = 'Ceci est une chaîne';
foo ($str);
echo $str;    // affiche 'Ceci est une chaîne'
foo (&$str);
echo $str;    // affiche 'Ceci est une chaîne, et un peu plus.'
```

[9.5.2.2 Valeur par défaut des arguments](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Vous pouvez définir comme en C++ des valeurs par défaut pour les arguments de type scalaire :

```
function servir_apero ($type = "ricard") {
    return "Servir un verre de $type.\n";
}
echo servir_apero();
echo servir_apero("whisky");
```

La fonction ci-dessus affichera :

Servir un verre de ricard.
Servir un verre de whisky.

La valeur par défaut d'un argument doit obligatoirement être une constante, et ne peut être ni une variable ou ni un membre de classe.

Il est à noter que vous utilisez les arguments par défaut, la valeur par défaut doit se trouver du côté droit du signe '='; sinon, cela ne fonctionnera pas. Considérons le code suivant :

```
<?php
function faireunyaourt ($type = "acidophilus", $flavour) {
return "Préparer un bol de $type $flavour.\n";
}
echo faireunyaourt ("framboise");    // ne fonctionne pas comme voulu
?>
```

L'affiche du code ci-dessus est le suivant :

```
Warning: Missing argument 2 in call to faireunyaourt() in
/usr/local/etc/httpd/htdocs/PHP 3test/functest.html on line 41
Préparer un bol de framboise.
```

Maintenant comparons l'exemple précédent avec l'exemple suivant :

```
<?php
function faireunyaourt ($flavour, $type = "acidophilus") {
return "Préparer un bol de $type $flavour.\n";
}
echo faireunyaourt ("framboise");    // fonctionne comme voulu
?>
```

L'affichage de cette exemple est le suivant :

```
Préparer un bol de acidophilus framboise.
```

9.5.2.3 Nombre d'arguments variable

[\[Notes en ligne\]](#) [\[Exemples\]](#)

PHP 4 supporte les fonctions à nombre d'arguments variable. C'est très simple à utiliser, avec les fonctions [func_num_args\(\)](#), [func_get_arg\(\)](#), et [func_get_args\(\)](#).

Aucune syntaxe particulière n'est nécessaire, et la liste d'argument doit toujours être fournie explicitement avec la définition de la fonction, et se comportera comme normalement.

9.5.3 Les valeurs de retour

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Les valeurs sont renvoyées en utilisant une instruction de retour optionnelle. Tous types de variables peuvent être renvoyées, tableaux et objets compris.


```
function carre ($num) {
return $num * $num;
}
echo carre (4);    // affiche '16'.
```

Vous ne pouvez pas renvoyer plusieurs valeurs en même temps, mais vous pouvez obtenir le même résultat en renvoyant un tableau.

```
function petit_nombre() {
return array (0, 1, 2);
}
list ($zero, $one, $two) = petit_nombre();
```

Pour retourner une référence d'une fonction,& aussi bien dans la déclaration de la fonction que dans l'assignation de la valeur de retour.

```
function &retourne_reference() {
return $uneref;
}
$newref =&retourne_reference();
```

9.5.4 old function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

L'instruction `old_function` vous permet de déclarer une fonction en utilisant une syntaxe du type PHP/FI2 (au détail près que vous devez remplacer l'instruction 'function' par 'old_function').

C'est une fonctionnalité obsolète et elle ne devrait être utilisée que dans le cadre de conversion de PHP/FI2 vers PHP 3

Les fonctions déclarées comme `old_function` ne peuvent pas être appelée à partir du code interne du PHP.

Cela signifie, par exemple, que vous ne pouvez pas les utiliser avec des fonctions comme [usort\(\)](#), [array_walk\(\)](#), et [register_shutdown_function\(\)](#). Vous pouvez contourner ce problème en écrivant une fonction d'encapsulation qui appellera la fonction `old_function`.

9.5.5 Variable functions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

PHP supporte le concept de fonctions variables. Cela signifie que si le nom d'une variable est suivi de parenthèses, PHP recherchera une fonction de même nom, et essaiera de l'exécuter. Cela peut servir, entre autre, lors pour faire des fonctions call-back, des tables de fonctions...

Exemple de fonction variable

```
<?php
function foo() {
echo "dans foo()<br>\n";
}
function bar( $arg = '' ) {
echo "Dans bar(); l'argument était '$arg'.<br>\n";
}
```

```
$func = 'foo';
$func();
$func = 'bar';
$func( 'test' );
?>
```

9.6 Classes et objets

[\[Notes en ligne\]](#)

9.6.1 Les classes : class

[\[Notes en ligne\]](#)

Une classe est une collection de variables et de fonctions qui fonctionnent avec ces variables. Une classe est définie en utilisant la syntaxe suivante :

```
<?php
class Cart {
var $items; // Eléments de notre panier
    // Ajout de $num articles de type $artnr au panier
function add_item ($artnr, $num) {
    $this->items[$artnr] += $num;
    }
    // Suppression de $num articles du type $artnr du panier
function remove_item ($artnr, $num) {
if ($this->items[$artnr] > $num) {
    $this->items[$artnr] -= $num;
return TRUE;
    } else {
return FALSE;
    }
    }
}
?>
```

L'exemple ci-dessus définit la classe Cart qui est composée d'un tableau associatif contenant les articles du panier et de deux fonctions, une pour ajouter et une pour enlever des éléments au panier.

Note : En PHP 4, seuls les initialiseurs constants pour les variables var sont autorisés. Utilisez les constructeurs pour les initialisation variables.

Les classes forment un type de variable. Pour créer une variable du type désiré, vous devez utiliser l'opérateur new.

```
<?php
$cart = new Cart;
$cart->add_item("10", 1);
?>
```

L'instruction ci-dessus crée l'objet \$cart de la class Cart. La fonction add_idem() est appelée afin d'ajouter l'article numéro 10 dans la panier.

Une classe peut être une extension d'une autre classe. Les classes "extended" ou "derived" héritent de toutes les variables et de toutes les fonctions de la classe père plus toutes les définitions que vous rajoutez à cette classe. Cela se fait avec le mot clef "extends". L'héritage multiple n'est pas supporté.

```
<?php
class Named_Cart extends Cart {
var $owner;
function set_owner ($name) {
    $this->owner = $name;
}
}
?>
```

L'exemple ci-dessus définit la classe `Named_Cart` qui possède les mêmes variables que la classe `Cart` et la variable `$owner` en plus, ainsi que la fonction `set_owner()`. Vous créez un panier nominatif de la même manière que précédemment, et vous pouvez alors affecter un nom au panier ou en connaître le nom. Vous pouvez de toutes les façons utiliser les mêmes fonctions que sur un panier classique.

```
<?php
$ncart = new Named_Cart;    // Création d'un panier nominatif
$ncart->set_owner ("kris"); // Affectation du nom du panier
print $ncart->owner;        // Affichage du nom du panier
$ncart->add_item ("10", 1); // (héritage des fonctions de la classe père)
?>
```

Dans les fonctions d'une classe, la variable `$this` est égale à l'objet de la classe. Vous pouvez utiliser la forme `"$this->quelquechose"` pour accéder aux fonctions ou aux variables de l'objet courant.

Le constructeur est la fonction qui est appelée automatiquement par la classe lorsque vous créez une nouvelle instance d'une classe. La fonction constructeur a le même nom que la classe.

```
<?php
class Auto_Cart extends Cart {
function Auto_Cart () {
    $this->add_item ("10", 1);
}
}
?>
```

L'exemple ci-dessus définit la classe `Auto_Cart` qui hérite de la classe `Cart` et définit le constructeur de la classe. Ce dernier initialise le panier avec 1 article de type numéro 10 dès que l'instruction "new" est appelée. La fonction constructeur peut prendre ou non, des paramètres optionnels, ce qui la rend beaucoup plus pratique.

```
<?php
class Constructor_Cart extends Cart {
function Constructor_Cart ($item = "10", $num = 1) {
    $this->add_item ($item, $num);
}
}
// Place dans le caddie toujours la même chose...
$default_cart = new Constructor_Cart;
// Place dans le caddie des objets différents, comme dans la
// réalité
$different_cart = new Constructor_Cart ("20", 17);
?>
```

Pour les classes qui utilisent l'héritage, le constructeur de la classe père n'est pas automatiquement appelé lorsque le constructeur de la classe dérivée est appelé.

9.7 Les opérateurs

[\[Notes en ligne\]](#)

9.7.1 Les opérateurs arithmétiques

[\[Notes en ligne\]](#)

Vous rappelez vous des opérations élémentaires apprises à l'école ?

Exemple	Nom	Résultat
$\$a + \b	Addition	Somme de \$a et \$b.
$\$a - \b	Soustraction	Différence de \$a et \$b.
$\$a * \b	Multiplication	Produit de \$a et \$b.
$\$a / \b	Division	Quotient de \$a et \$b.
$\$a \% \b	Modulo	Reste de \$a divisé par \$b.

L'opérateur de division ("/") retourne une valeur entière (le résultat d'une division entière) si les deux opérandes sont entiers (ou bien des chaînes converties en entiers. Si l'un des opérandes est un nombre à virgule flottante, ou bien le résultat d'une opération qui retourne une valeur non entière, un nombre à virgule flottante sera retourné.

9.7.2 Les opérateurs d'assignement

[\[Notes en ligne\]](#)

L'opérateurs d'assignement le plus simple est le signe "=". Le premier réflexe est de penser que ce signe veut dire "égal à". Ce n'est pas le cas. Il signifie que l'opérande de gauche se voit affecter la valeur de l'expression qui est à droite du signe égal.

La valeur d'une expression d'assignement est la valeur assignée. Par exemple, la valeur de l'expression '\$a = 3' est la valeur 3. Cela permet de faire d'utiliser des astuces telles que :

```
<?php
$a = ($b = 4) + 5; // $a est maintenant égal à 9, et $b vaut 4.
?>
```

En plus du simple opérateur d'assignement, il existe des "opérateurs combinés" pour tous les opérateurs arithmétiques et pour les opérateurs sur les chaînes de caractères. Cela permet d'utiliser la valeur d'une variable dans une expression et d'affecter le résultat de cette expression à cette variable. Par exemple:

```
<?php
$a = 3;
$a += 5;
// affecte la valeur 8 à la variable $a.
// correspond à l'instruction '$a = $a + 5');
$b = "Bonjour ";
$b .= " tout le monde!";
// affecte la valeur "Bonjour tout le monde!" à
```

```
la variable $b (correspond à $b = $b." tout le monde!";
?>
```

On peut noter que l'assignement copie le contenu de la variable originale dans la nouvelle (assignement par valeur), ce qui fait que les changements de valeur d'une variable ne modifieront pas la valeur de l'autre. Cela peut se révéler important lors de la copie d'un grand tableau durant une boucle. PHP 4 supporte aussi l'assignement par référence, en utilisant la syntaxe `$var = &$othervar`; mais ce n'était pas possible en PHP 3. 'L'assignement par référence' signifie que les deux variables contiennent les mêmes données, et que la modification de l'une affecte l'autre. D'un autre côté, la recopie est très rapide.

9.7.3 Bitwise Operators

[\[Notes en ligne\]](#)

Les opérateurs sur les bits vous permettent de manipuler les bits dans un entier.

exemple	nom	résultat
<code>\$a & \$b</code>	ET (AND)	Les bits positionnés à 1 dans \$a ET dans \$b sont positionnés à 1. @tab
<code>\$a \$b</code>	OU (OR)	Les bits positionnés à 1 dans \$a OU \$b sont positionnés à 1. @tab
<code>\$a ^ \$b</code>	Xor	Les bits positionnés à 1 dans \$a OU dans \$b sont positionnés à 1. @tab
<code>~ \$a</code>	NON (Not)	Les bits qui sont positionnés à 1 dans \$a sont positionnés à 0, et vice versa. @tab
<code>\$a << \$b</code>	Décalage à gauche	Décale les bits de \$a dans \$b par la gauche (chaque décalage équivaut à une multiplication par 2). @tab
<code>\$a >> \$b</code>	Décalage à droite	Décalage des bits de \$a dans \$b par la droite (chaque décalage équivaut à une division par 2). @tab

9.7.4 Opérateurs de comparaison

[\[Notes en ligne\]](#)

Les opérateurs de comparaison, comme leur nom l'indique, vous permettent de comparer deux valeurs.

Exemple	Nom	Résultat
<code>\$a == \$b</code>	Egal	Vrai si \$a est égal à \$b.
<code>\$a === \$b</code>	Identique	Vrai si \$a est égal à \$b et qu'ils sont de même type (PHP 4 seulement).

		@tab
<code>\$a != \$b</code>	Différent	Vrai si \$a est différent de \$b.
<code>\$a < \$b</code>	Plus petit que	Vrai si \$a est plus petit strictement que \$b.
<code>\$a > \$b</code>	Plus grand	Vrai si \$a est plus grand strictement que \$b.
<code>\$a <= \$b</code>	Inférieur ou égal	Vrai si \$a est plus petit ou égal à \$b.
<code>\$a >= \$b</code>	Supérieur ou égal	Vrai si \$a est plus grand ou égal à \$b.

Un autre opérateur conditionnel est l'opérateur ternaire (":?"), qui fonctionne comme en langage C.

```
<?php
(expr1) ? (expr2) : (expr3);
?>
```

Cette expression renvoie la valeur de l'expression *expr2* si l'expression *expr1* est vraie, et l'expression *expr3* si l'expression *expr1* est fausse.

9.7.5 Opérateur de contrôle d'erreur

[\[Notes en ligne\]](#)

PHP supporte un opérateur de contrôle d'erreur : c'est `@`. Lorsque cet opérateur est ajouté en préfixe d'une expression PHP, les messages d'erreur qui peuvent être générés par cette expression seront ignorés. Si l'option [7.1.1.26 ini.track_errors](#) est activée, les messages d'erreurs générés une expression seront sauvés dans la variable globale `$php_errormsg`. Cette variable sera écrasée à chaque erreur. Il faut alors la surveiller souvent pour pouvoir l'utiliser.

```
<?php
/* Erreur SQL intentionnelle (trop de guillemets): */
$res = @mysql_query( "select nom, code from 'listedenom" ) ou
die( "La requête a échoué : l'erreur est '$php_errormsg' " );
?>
```

Voir aussi [error_reporting\(\)](#).

9.7.6 Opérateur d'exécutions

[\[Notes en ligne\]](#)

PHP supporte un opérateur d'exécution : guillemets obliques ("``"). Notez bien la différence avec les guillemets simples (sur la touche 4), et ceux-ci (sur la touche de la livre anglaise). PHP essaiera d'exécuter le contenu de ces guillemets obliques comme une commande shell. Le résultat sera retourné (i.e. : il ne sera pas simplement envoyé à la sortie standard, il peut être assigné à une variable).

```
<?php
$output = `ls -al`;
echo "<pre>$output</pre>";
?>
```

Note : Cet opérateur est désactivé lorsque le [7.1.3.1 ini.safe_mode](#) est activé.

Voir aussi [system\(\)](#), [passthru\(\)](#), [exec\(\)](#), [popen\(\)](#), et [escapeshellcmd\(\)](#).

9.7.7 Opérateurs d'incrementation/Décrementation

[\[Notes en ligne\]](#)

PHP supporte les opérateurs de pré et post incrémentation et décrémentation, comme en C.

Exemple	Nom	Résultat
<code>++\$a</code>	Pré-incrémente	Incrémente \$a de 1, puis retourne \$a.
<code>\$a++</code>	Post-incrémente	Retourne \$a, puis l'incrémente \$a de 1.
<code>--\$a</code>	Pré-décrémente	Décrémente \$a de 1, puis retourne \$a.
<code>\$a--</code>	Post-décrémente	Retourne \$a, puis décrémente \$a de 1.

Voici un exemple simple

```
<?php
echo "<h3>Post-incrémentation</h3>";
$a = 5;
echo "Devrait valoir 5: " . $a++ . "<br>\n";
echo "Devrait valoir 6: " . $a . "<br>\n";
echo "<h3>Pré-incrémentation</h3>";
$a = 5;
echo "Devrait valoir 6: " . ++$a . "<br>\n";
echo "Devrait valoir 6: " . $a . "<br>\n";
echo "<h3>Post-décrémentation</h3>";
$a = 5;
echo "Devrait valoir 5: " . $a-- . "<br>\n";
echo "Devrait valoir 4: " . $a . "<br>\n";
echo "<h3>Pré-décrementation</h3>";
$a = 5;
echo "Devrait valoir 4: " . --$a . "<br>\n";
echo "Devrait valoir 4: " . $a . "<br>\n";
?>
```

9.7.8 Les opérateurs logiques

[\[Notes en ligne\]](#)

Exemple	Nom	Résultat
<code>\$a and \$b</code>	ET (And)	Vrai si \$a ET \$b sont vrais.
<code>\$a or \$b</code>	OU (Or)	Vrai si \$a OU \$b est vrai
<code>\$a xor \$b</code>	XOR (Xor)	Vrai si \$a OU \$b est vrai, mais pas les deux en même temps.
<code>! \$a</code>	NON (Not)	Vrai si \$a est faux.
<code>\$a && \$b</code>	ET (And)	Vrai si \$a ET \$b sont vrais.
<code>\$a \$b</code>	OU (Or)	Vrai si \$a OU \$b est vrai.

La raison pour laquelle il existe deux types de "ET" et de "OU" est qu'ils ont des priorités différentes. Voir le

paragraphe [9.7.9 La précedence des opérateurs](#).

9.7.9 La précedence des opérateurs

[\[Notes en ligne\]](#)

La priorité des opérateurs spécifie l'ordre dans lequel les valeurs doivent être analysées. Par exemple, dans l'expression `1 + 5 * 3`, le résultat est 16 et non 18, car la multiplication ("`*`") a une priorité supérieure par rapport à l'addition ("`+`").

Le tableau suivant dresse une liste de la priorité des différents opérateurs dans un ordre croissant de priorité.

Associativité	Opérateurs
gauche	,
gauche	or
gauche	xor
gauche	and
droite	print
gauche	= += -- *= /= .= %= = ^= ~= <=>>=
gauche	? :
gauche	
gauche	&&
gauche	
gauche	^
gauche	&
non-associative	== != ===
non-associative	< <= > >=
gauche	<< >>
gauche	+ - .
gauche	* / %
droite	! ~ ++ -- (int) (double) (string) (array) (object)
droite	[
non-associative	new

9.7.10 Opérateurs de chaînes

[\[Notes en ligne\]](#)

Il y a deux opérateurs de chaînes. Le premier est l'opérateur de concaténation ("`.`"), qui retourne la concaténation de ses deux arguments. Le second est l'opérateur d'assignement concaténant ("`.=`"). Reportez vous à [9.7.2 Les opérateurs d'assignement](#) pour plus de détails.


```
<?php
$a = "Bonjour ";
$b = $a . "Monde!"; // $b contient "Bonjour Monde!"
$a = "Bonjour ";
$a = $a . "Monde!"; // $a contient "Bonjour Monde!"
?>
```

9.8 Types

[\[Notes en ligne\]](#)

PHP supporte les types suivants :

- [9.8.4 Les tableaux](#)
- [9.8.2 Les nombres à virgule flottante](#)
- [9.8.1 Entiers](#)
- [9.8.5 Les objets](#)
- [9.8.3 Les chaînes de caractères](#)

Habituellement, le type d'une variable n'est pas déclaré par le programmeur. Il est décidé au moment de l'exécution par le PHP, en fonction du contexte dans lequel la variable est utilisée.

Si vous voulez forcer une variable à être convertie en un certain type, vous devez transtyper ([9.8.6.1 Transtypage](#)) la variable ou utiliser la fonction [settype\(\)](#).

Il est à noter qu'une variable peut se comporter de manière différente suivant les situations, en fonction du type qui lui est affecté. Pour plus d'informations, voir le paragraphe [9.8.6 Définition du type](#).

9.8.1 Entiers

[\[Notes en ligne\]](#)

Il est possible de spécifier les nombres entiers (integers) de la manière suivante :

```
<?php
$a = 1234; # nombre entier en base 10
$a = -123; # nombre entier négatif
$a = 0123; # nombre entier en base 8, octale (équivalent à 83 en base 10)
$a = 0x12; # nombre entier en base 16, hexadécimale (équivalent à 18 en base 10)
?>
```

9.8.2 Les nombres à virgule flottante

[\[Notes en ligne\]](#)

Les nombres à virgule flottante ("double") peuvent être spécifiés en utilisant la syntaxe suivante:

```
<?php
$a = 1.234;
$a = 1.2e3;
?>
```

Il est fréquent que de simples fractions décimales telles que 0.1 ou 0.7 ne puissent être converties au format interne binaire sans une légère perte de précision. Cela peut conduire à des résultats étonnants : par exemple, `floor((0.1+0.7)*10)` retournera 7 au lieu de 8 car le résultat de la représentation interne est 7.999999999.... Tout ceci est lié au fait qu'il est impossible d'exprimer certaines fractions en un nombre fini de chiffres. Par exemple 1/3 s'écrira 0.3333333... en mode décimal.

Ne faites donc jamais confiance aux nombres à virgule jusqu'à leur dernière décimale, et ne comparer jamais ces nombres avec l'opérateur d'égalité. Si vous avez besoin d'une précision particulière, reportez vous au traitement des nombres de grande taille avec les librairies [10.4 Nombres de grande taille](#) ou [10.26 GMP](#).

9.8.3 Les chaînes de caractères

[\[Notes en ligne\]](#)

Les chaînes de caractères peuvent être définies en utilisant deux types de délimiteurs.

Si la chaîne de caractères est délimitée par des guillemets doubles ("), les variables à l'intérieur de la chaîne seront évaluées, et remplacées par leur valeur. Comme en C ou en Perl, le caractère antislash (\) est utilisé pour protéger (échapper) un caractère spécial.

séquence	valeur
<code>\n</code>	Nouvelle ligne (linefeed, LF ou 0x0A en ASCII)
<code>\r</code>	Retour à la ligne(carriage return, CR ou 0x0D en ASCII)
<code>\t</code>	Tabulation horizontale (HT ou 0x09 en ASCII)
<code>\\</code>	Antislash
<code>\\$</code>	Caractère \$
<code>\"</code>	Guillemet double
<code>\[0-7]{1,3}</code>	Une séquence de caractère qui permet de rechercher un nombre en notation octale. @tab
<code>\x[0-9A-Fa-f]{1,2}</code>	Une séquence de caractère qui permet de rechercher un nombre en notation hexadécimale. @tab

Vous pouvez utiliser le caractère d'échappement antislash sur n'importe quel autre caractère, mais cela produira une alerte (si le niveau d'alerte maximal a été fixé). En PHP 3.0, une alerte sera affichée, si le niveau de rapport d'erreur est à E_NOTICE. En PHP 4.0, aucune alerte ne sera générée.

Le deuxième moyen de délimiter une chaîne de caractère est d'utiliser les guillemets simples (""). Dans une telle chaîne de caractères, les variables *ne seront pas* évaluées, et le caractère antislash n'aura aucun effet (à deux exceptions près, pour `\"` et `\\`, afin de pouvoir utiliser les caractères guillemets simples, et antislash dans la chaîne de caractères).

Un autre moyen de délimiter les chaînes est d'utiliser la syntaxe de "Here doc" (en français, doc ici): `<<<`, suivi d'un identifiant arbitraire, puis de la chaîne. Cette séquence se termine par l'identifiant initial, placé en premier sur une nouvelle ligne. L'identifiant utilisé doit suivre les mêmes règles que les étiquettes PHP : il ne

doit contenir uniquement des caractères alpha-numériques, et des soulignés ("_"), et enfin, commencer par un caractère alphabétique ou un souligné.

La syntaxe Here doc se comporte exactement comme une chaîne à guillemets doubles, sans les guillemets doubles. Cela signifie que vous n'avez pas à échapper les guillemets (simples ou doubles) dans cette syntaxe. Les variables sont remplacées par leur valeur, et le même soin doit leur être apporté que dans les chaînes à guillemets doubles.

Exemple de chaîne Here Doc

```
<?php
$str = <<<EOD
Exemple de chaîne
s'étalant sur
plusieurs lignes
avec la syntaxe heredoc
EOD;
/* Exemple plus complexe, avec des variables. */
class foo {
var $foo;
var $bar;
function foo() {
    $this->foo = 'Foo';
    $this->bar = array('Bar1', 'Bar2', 'Bar3');
}
}
$foo = new foo();
$name = 'MonNom';
echo <<<EOT
Mon nom est "$name". J'affiche des $foo->foo.
Maintenant, j'affiche un {$foo->bar[1]}.
Ceci se traduit par un 'A' majuscule: \x41
EOT;
?>
```

Note : Le support Here doc a été ajouté en PHP 4.

Les chaînes peuvent être concaténées avec l'opérateur '.' (point). Notez que l'opérateur '+' (addition) ne fonctionnera pas. Reportez vous à [9.7.10 Opérateurs de chaînes](#) pour plus d'informations.

Quelques exemples de chaînes

```
<?php
/* Assignment d'une chaîne */
$str = "Ceci est une chaîne ";
/* Concaténation d'une chaîne */
$str = $str . " avec une extension";
/* Une autre méthode de concaténation, y compris une nouvelle ligne */
$str .= " et terminée par une nouvelle ligne.\n";
/* Cette chaî se terminera par '<B>Nombre: 9</B>' */
$nombre = 9;
$str = "<B>Nombre: $nombre</B>";
/* Cette ci sera '<B>Nombre: $num</B>' */
$num = 9;
$str = '<B>Nombre: $num</B>';
/* Lire le premier caractère d'une chaîne */
$str = 'Ceci est un test.';
$first = $str[0];
/* Lire le dernier caractère d'une chaîne */
$str = 'Ceci est un autre test.';
$last = $str[strlen($str)-1];
```

?>

9.8.3.1 Conversion de type

[\[Notes en ligne\]](#)

Lorsqu'une chaîne de caractère est évaluée comme une valeur numérique, le résultat et le type de la variable sont déterminés comme suit.

La chaîne de caractères est de type "double" si elle contient un des caractère '.', 'e' ou 'E'. Sinon, elle est de type entier ("integer").

La valeur est définie par la première partie de la chaîne. Si la chaîne de caractères débute par une valeur numérique cette valeur sera celle utilisée. Sinon, la valeur sera égale à 0 (zéro).

Lorsque la première expression est une chaîne de caractères, le type de la variable dépend de la seconde expression.

```
<?php
$foo = 1 + "10.5";           // $foo est du type double (11.5)
$foo = 1 + "-1.3e3";         // $foo est du type double (-1299)
$foo = 1 + "bob-1.3e3";      // $foo est du type integer (1)
$foo = 1 + "bob3";           // $foo est du type integer (1)
$foo = 1 + "10 Small Pigs";  // $foo est du type integer (11)
$foo = 1 + "10 Little Piggies"; // $foo est du type integer (11)
$foo = "10.0 pigs " + 1;      // $foo est du type integer (11)
$foo = "10.0 pigs " + 1.0;    // $foo est du type double (11)
?>
```

Pour plus d'informations sur les conversions de type, voir les pages de man à propos de la fonction strtod(3).

Si vous voulez tester l'un des exemples de cette section, vous pouvez faire un copier/coller et l'insérer dans un script pour voir comment il se comporte.

```
<?php
echo "\$foo==\$foo; type is " . gettype( $foo ) . "<br>\n";
?>
```

9.8.4 Les tableaux

[\[Notes en ligne\]](#)

Les tableaux ressemblent aux tables de hashage (tableaux associatifs) et aux tableaux indexés (vecteurs).

9.8.4.1 Tableaux à une dimension

[\[Notes en ligne\]](#)

PHP supporte les tableaux scalaires et les tableaux associatifs. En fait, il n'y a aucune différence entre les deux. Vous pouvez créer un tableau en utilisant les fonctions [list\(\)](#) ou [array\(\)](#), ou bien en affectant explicitement chacune des valeurs.

```
<?php
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
?>
```

Vous pouvez aussi créer un tableau en ajoutant simplement les valeurs à ce tableau.

```
<?php
$a[] = "Bonjour"; // $a[2] == "Bonjour";
$a[] = "Monde"; // $a[3] == "Monde";
?>
```

Un tableau peut être trié en utilisant les fonctions [asort\(\)](#), [arsort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), [sort\(\)](#), [uasort\(\)](#), [usort\(\)](#), ou [uksort\(\)](#) en fonction du type de classement que vous voulez.

Vous pouvez compter le nombre d'éléments qu'il y a dans un tableau en utilisant la fonction [count\(\)](#).

Vous pouvez vous déplacer à l'intérieur d'un tableau en utilisant les fonctions [next\(\)](#) et [prev\(\)](#). Un autre moyen de se déplacer dans un tableau est d'utiliser la fonction [each\(\)](#).

9.8.4.2 Tableaux à plusieurs dimensions

[\[Notes en ligne\]](#)

Les tableaux à plusieurs dimensions sont extrêmement simples. Pour chaque dimension du tableau, vous ajouter une nouvelle [clef] à la fin:

```
<?php
$a[1]          = $f;           # tableau à une dimension
$a["foo"]      = $f;
$a[1][0]       = $f;           # tableau à deux dimensions
$a["foo"][2]   = $f;           # vous pouvez mélanger les indices associatifs et numériques
$a[3]["bar"]   = $f;           # vous pouvez mélanger les indices associatifs et numériques
$a["foo"][4]["bar"][0] = $f;   # tableau à quatre dimensions
?>
```

En PHP 3 il n'est pas possible de référencer un tableau à l'intérieur d'une chaîne. Par exemple, ceci ne fonctionne pas :

```
<?php
$a[3]['bar'] = 'Bob';
echo "Cela ne marche pas : $a[3][bar]";
?>
```

En PHP 3, l'exemple ci dessus va afficher : Cela ne marche pas : Array[bar]. L'opérateur de concaténation, peut être utilisé pour corriger cela :

```
<?php
$a[3]['bar'] = 'Bob';
echo "Cela ne marche pas : " . $a[3][bar];
?>
```

En PHP 4, cependant, le problème peut être contourné en entourant le tableau par des accolades :

```
<?php
$a[3]['bar'] = 'Bob';
echo "Cela marche : {$a[3][bar]}";
?>
```

Vous pouvez remplir un tableau à plusieurs dimensions par de nombreux moyens mais la méthode la plus simple à comprendre est l'utilisation de la fonction [array\(\)](#). Les deux exemples suivants montrent comment remplir un tableau à une dimension:

```
<?php
# Exemple 1:
$a["couleur"]    = "rouge";
$a["saveur"]     = "sucrée";
$a["forme"]      = "rond";
$a["nom"]        = "pomme";
$a[3]            = 4;
# Exemple 2:
$a = array(
    "couleur" => "rouge",
    "saveur"  => "sucrée",
    "forme"   => "rond",
    "nom"     => "pomme",
    3         => 4
);
?>
```

La fonction [array\(\)](#) peut être emboîtée pour remplir un tableau à plusieurs dimensions :

```
<?php
$a = array(
    "pomme" => array(
        "couleur" => "rouge",
        "saveur"  => "sucrée",
        "forme"   => "rond"
    ),
    "orange" => array(
        "couleur" => "orange",
        "saveur"  => "amère",
        "forme"   => "rond"
    ),
    "banane" => array(
        "couleur" => "jaune",
        "saveur"  => "paste-y",
        "forme"   => "bananoïde"
    )
);
echo $a["pomme"]["saveur"];    # va afficher "sucrée"
?>
```

9.8.5 Les objets

[\[Notes en ligne\]](#)

9.8.5.1 Initialisation d'un objet

[\[Notes en ligne\]](#)

Pour initialiser un objet, vous devez utiliser la commande "new" afin de créer l'instance de l'objet.

```
<?php
```

```

class foo {
function faire_foo () {
echo "Faisant foo.";
}
}
$bar = new foo;
$bar->do_foo();
?>

```

9.8.6 Définition du type

[\[Notes en ligne\]](#)

PHP ne nécessite pas de déclaration explicite du type d'une variable. Le type d'une variable est déterminé par le contexte d'utilisation. Par exemple, si vous assignez une chaîne de caractères à la variable *var*, var devient une chaîne de caractère. Si vous assignez un nombre entier à *var*, elle devient un entier.

Un exemple de convertisseur automatique de type est l'opérateur '+'. Si un des opérandes est de type double, alors tous les opérandes sont évalués comme des variables de type double et le résultat est de type double. Sinon, tous les opérandes sont évalués comme des variables de type entier et le résultat sera du type entier. Il est à noter que cela NE CHANGE PAS le type des opérandes. Le seul changement est la manière dont les opérandes sont évaluées.

```

<?php
$foo = "0"; // $foo est une chaîne de caractères (ASCII 48)
$foo++; // $foo est la chaîne de caractères "1" (ASCII 49)
$foo += 1; // $foo est maintenant du type entier (2)
$foo = $foo + 1.3; // $foo est maintenant du type double (3.3)
$foo = 5 + "10 Petits cochons"; // $foo est du type entier (15)
$foo = 5 + "10 cochonnets"; // $foo est du type entier (15)
?>

```

Si les deux derniers exemples vous semblent obscurs ou si vous voulez forcer une variable à être évaluée avec un certain type, reportez vous au paragraphe Conversion de type ci-dessous. Si vous voulez changer le type d'une variable, intéressez vous à aux fonctions de [9.8.3.1 Conversion de type](#).

Si vous voulez forcer le type d'une variable, vous pouvez vous reporter à la section [9.8.6.1 Transtypage](#). Si vous voulez changer le type d'une variable, utilisez [settype\(\)](#).

Pour tester les exemples de cette section, il suffit d'en faire un copier/coller, et d'insérer les lignes dans un script PHP.

```

<?php
echo "\$foo==\$foo; le type est " . gettype( $foo ) . "<br>\n";
?>

```

Note : Le comportement de la conversion automatique est actuellement indéfinie.

```

<?php
$a = 1; // $a est un entier
$a[0] = "f"; // $a devient un tableau, et $a[0] contient "f"
?>

```

Bien que dans l'exemple ci dessus, il semble évident que \$a va devenir un tableau, dont le premier élément

est 'f', considérez l'exemple suivant :

```
<?php
$a = "1";      // $a est une chaîne
$a[0] = "f";   // Qu'est ce qu'une position dans une chaîne ? que se passe t il?
?>
```

Etant donné que PHP supporte l'indexation de chaîne avec des offsets identiques à celles des tableaux, l'exemple ci dessus conduit à un problème : est ce que \$a est un tableau, dont le premier élément est "f", ou bien est ce que "f" est le premier élément de la chaîne de caractères \$a?

Pour cette raison, aussi bien pour PHP 3.0.12 que PHP 4.0b3–RC4, le résultat de la conversion automatique est considéré comme indéfinie. Des solutions sont en cours de discussion.

9.8.6.1 Transtypage

[\[Notes en ligne\]](#)

La conversion de type en PHP fonctionne de la même manière qu'en C: le nom du type désiré est écrit entre parenthèses devant la variable à transtyper ("cast").

```
<?php
$foo = 10;     // $foo est un entier
$bar = (double) $foo; // $bar est un double
?>
```

Les conversions autorisées sont:

- (int), (integer) – type entier
- (real), (double), (float) – type double
- (string) – ctype chaîne
- (array) – type tableau
- (object) – type objet

Il est à noter que les tabulations et les espaces sont autorisés à l'intérieur des parenthèses, donc les lignes suivantes sont équivalentes:

```
<?php
$foo = (int) $bar;
$foo = ( int ) $bar;
?>
```

Le transtypage n'a pas toujours le résultat attendu. Par exemple :

Lorsque vous transtypez un scalaire ou une chaîne en tableau, la variable verra son contenu affecté au premier élément du tableau.

```
<?php
$var = 'ciao';
$arr = (array) $var;
echo $arr[0]; // affiche 'ciao'
?>
```


Lorsque vous transtypez un scalaire ou une chaîne en objet, la valeur de la variable sera transformée en attribut de l'objet. L'attribut s'appellera 'scalar':

```
<?php
$var = 'ciao';
$obj = (object) $var;
echo $obj->scalar; // affiche 'ciao'
?>
```

< @node language.variables , language.types.typecasting, language.variables.basics, Top

9.9 Les variables

[\[Notes en ligne\]](#)

9.9.1 Essentiel

[\[Notes en ligne\]](#)

En PHP, les variables sont représentées par un signe dollar "\$" suivi du nom de la variable. Le nom est sensible à la casse (ie : \$x != \$X).

Les noms de variables suivent les mêmes règles de nommage que les autres entités PHP. Un nom de variable valide doit commencer par une lettre ou un souligné (_), suivi de lettres, chiffres ou soulignés. Exprimé sous forme d'expressions régulière, cela donne : '[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*'

Note : Dans nos propos, une lettre est a-z, A-Z, et les caractères ASCII de 127 à 255 (0x7f-0xff).

```
<?php
$var = "Jean";
$Var = "Paul";
echo "$var, $Var"; // affiche "Jean, Paul"
$4site = 'pas encore'; // invalide : commence par un nombre
$_4site = 'pas encore'; // valide : commence par un souligné
$täyte = 'mansikka'; // valid; 'ä' is ASCII 228.
?>
```

En PHP3, les variables sont toujours assignées par valeur. C'est à dire, lorsque vous assignez une expression à une variable, la valeur de l'expression est recopiée dans la variable. Cela signifie, par exemple, qu'après avoir assigné la valeur d'une variable à une autre, modifier une variable n'aura pas d'effet sur l'autre. Pour plus de détails sur ce genre d'assignement, reportez vous à [9.4 Les expressions](#).

PHP 4.0 permet aussi d'assigner les valeurs aux variables *par référence*. Cela signifie que la nouvelle variable ne fait que référencer (en d'autres terme, "devient un alias de", ou encore "pointe sur") la variable originale. Les modifications de la nouvelle variable affecteront l'ancienne, et vice versa. Cela signifie aussi qu'aucune copie n'est faite : l'assignement est donc beaucoup plus rapide. Cela se fera notamment sentir dans des boucles, ou lors d'assignement de grands objets (tableaux).

Pour assigner par référence, ajoutez simplement un &(ET commercial) au début de la variable qui est assignée (la variable source). Dans l'exemple suivant, 'Mon nom est Pierre' s'affichera deux fois :

```
<?php
$foo = 'Pierre'; // Assigne la valeur 'Pierre' à $foo
$bar = &$foo; // Référence $foo avec $bar.
$bar = "Mon nom est Pierre"; // Modifie $bar...
```

```
echo $foo;           // $foo est aussi modifiée
echo $bar;
?>
```

Une chose importante à noter est que seules, les variables nommées peuvent être assignées par référence.

```
<?php
$foo = 25;
$bar = &$foo;      // Assignment valide .
$bar = &(24 * 7);  // Assignment Invalide : référence une expression sans nom
function test() {
    return 25;
}
$bar = &test();    // Invalide.
?>
```

9.9.2 Variables prédéfinies

[\[Notes en ligne\]](#)

PHP fournit un grand nombre de variables prédéfinies. Cependant, beaucoup de ces variables ne peuvent pas être présentées ici, car elles dépendent du serveur sur lequel elles tournent, de la version du serveur, et de la configuration du serveur, ou encore d'autres facteurs.. Certaines de ces variables ne seront pas accessibles lorsque PHP fonctionne en exécutable.

Malgré ces données, voici une liste de variables prédéfinies, qui seront accessibles avec une installation ad hoc de PHP3, fonctionnant en module, sous [Apache](#) 1.3.6.

Pour la liste complète des variables prédéfinies (et d'autres informations pratiques) reportez vous (et usez) de [phpinfo\(\)](#).

Note : Cette liste n'est pas exhaustive et ne le sera pas. C'est simplement un aperçu des variables prédéfinies qui peuvent être accessibles dans les scripts.

9.9.2.1 Variables Apache

[\[Notes en ligne\]](#)

Ces variables sont créées par le serveur [Apache](#). Si vous utilisez un autre serveur web, il n'est pas sûr que celui-ci vous fournira les mêmes variables. Il peut ne pas les fournir, en fournir d'autres. Cependant, un bon nombre de ces variables font partie de l'interface [CGI 1.1](#), et on peut s'attendre à les retrouver.

Notez que peu d'entre elles seront accessibles lorsque PHP est appelé en ligne de commande, (et elles n'auront alors peut-être pas de sens)

GATEWAY_INTERFACE

- Numéro de révision de l'interface CGI du serveur : i.e. 'CGI/1.1'.
SERVER_NAME
- Le nom du serveur hôte qui exécute le script suivant. Si le script est exécuté sur un hôte virtuel, ce sera la valeur définie pour cet hôte virtuel.
SERVER_SOFTWARE
- Chaîne d'identification du serveur, qui est donnée dans les entêtes lors de la réponse aux requêtes.
SERVER_PROTOCOL
- Nom et révision du protocole de communication : i.e. 'HTTP/1.0';
REQUEST_METHOD

- Méthode de requête utilisée pour accéder à la page; i.e. 'GET', 'HEAD', 'POST', 'PUT'.
QUERY_STRING
- La chaîne de requête, si elle existe, qui est utilisée pour accéder à la page.
DOCUMENT_ROOT
- La racine sous laquelle le script courant est exécuté, comme défini dans la configuration du serveur.
HTTP_ACCEPT
- Contenu de l'entête Accept: de la requête courant, si il y en a une.
HTTP_ACCEPT_CHARSET
- Contenu de l'entête Accept-Charset: de la requête courante, si il existe. Par exemple :
'iso-8859-1,*,utf-8'.
HTTP_ENCODING
- Contenu de l'entête Accept-Encoding: de la requête courante, si elle existe. Par exemple : 'gzip'.
HTTP_ACCEPT_LANGUAGE
- Contenu de l'entête Accept-Language: de la requête courante, si elle existe. Par exemple : 'en'.
HTTP_CONNECTION
- Contenu de l'entête Connection: de la requête courante, si elle existe. Par exemple : 'Keep-Alive'.
HTTP_HOST
- Contenu de l'entête Host: de la requête courante, si elle existe.
HTTP_REFERER
- L'adresse de la page (si elle existe) qui a conduit le client à la page courante. Cette valeur est affectée par le client, et tous les clients ne le font pas.
HTTP_USER_AGENT
- Contenu de l'entête User-Agent: de la requête courante, si elle existe. C'est une chaîne qui décrit le client HTML utilisé pour voir la page courante. Par exemple : Mozilla/4.5 [en] (X11; U; Linux 2.2.9 i586). Entre autres choses, vous pouvez utiliser cette valeur avec [get_browser\(\)](#) pour optimiser votre page en fonction des capacité du client.
REMOTE_ADDR
- L'adresse IP du client qui demande la page courante.
REMOTE_PORT
- Le port utilisé par la machine cliente pour communiquer avec le serveur web.
SCRIPT_FILENAME
- Le chemin absolu jusqu'au script courant.
SERVER_ADMIN
- La valeur donné à la directive SERVER_ADMIN (pour Apache), dans le fichier de configuration. Si le script est exécuté par un hôte virtuel, cela sera la valeur définie par l'hôte virtuel.
SERVER_PORT
- Le port de la machine serveur utilisé pour les communications. Par défaut, c'est '80'; en utilisant SSL, par exemple, il sera remplacé par le numéro de port HTTP sécurisé.
SERVER_SIGNATURE
- Chaîne contenant le numéro de version du serveur et le nom d'hôte virtuel, qui sont ajoutés aux pages générées par le serveur, si cette option est activée.
PATH_TRANSLATED
- Chemin dans le système de fichier (pas le document root-) jusqu'au script courant, une fois que le serveur à fait une chemin traduction virtuel->réel.
SCRIPT_NAME
- Contient le nom du script courant. Cela sert lorsque les pages doivent s'appeler elles-mêmes.
REQUEST_URI
- L'URI qui a été fourni pour accéder à cette page. Par exemple : '/index.html'.

9.9.2.2 Variables d'environnement

[\[Notes en ligne\]](#)

Ces variables sont importées dans l'espace de nom global de PHP, depuis l'environnement sous lequel PHP fonctionne. Beaucoup d'entre elles sont fournies par le shell qui exécute PHP et différents systèmes étant susceptibles de disposer de différents shells, une liste définitive est impossible à établir. Reportez vous à la documentation de votre shell, pour connaître la liste des variables d'environnement prédéfinies.

Les autres variables d'environnements incluent les variables CGI, placées ici, quelque fois la méthode d'exécution de PHP (CGI ou module).

9.9.2.3 Variables PHP

[\[Notes en ligne\]](#)

Ces variables sont créées par PHP lui-même. Les variables `$HTTP_*_VARS` ne sont disponibles que si l'option de configuration [7.1.1.27 ini.track-vars](#) a été activée. Lorsque c'est le cas, ces variables existent toujours, même si ce sont des tableaux vides. Cela évite les usurpations malintentionnées de ces variables.

Note : Depuis PHP 4.0.3, [7.1.1.27 ini.track-vars](#) est toujours activé, quelque soit la configuration.

Si la directive [7.1.1.23 ini.register-globals](#) est activée, alors ces variables seront aussi disponibles comme variable global du script : c'est à dire, indépendamment des tableaux `$HTTP_*_VARS`. Cette fonctionnalité doit être utilisée avec précautions, et de préférence, désactivée. Si `$HTTP_*_VARS` est sécurisé, les équivalents globaux peuvent être écrasés par les données d'entrée de l'utilisateur, avec des intrusions possibles. Si vous ne pouvez pas désactiver [7.1.1.23 ini.register-globals](#), vous devez prendre toutes les dispositions possibles pour vous assurer que les données utilisées sont sûres.

`argv`

- Tableau des arguments passés au script. Lorsque le script est appelé en ligne de commande, cela donne accès aux arguments, comme en langage C. Lorsque le script est appelé avec la méthode GET, ce tableau contiendra la chaîne de requête.

`argc`

- Contient le nombre de paramètres de la ligne de commande passés au script (s'il fonctionne en ligne de commande).

`PHP_SELF`

- Le nom du fichier du script en cours d'exécution, par rapport au document root. Si PHP fonctionne en ligne de commande, cette variable n'est pas disponible.

`HTTP_COOKIE_VARS`

- Un tableau associatif des variables passées au script courant via les HTTP cookies. Uniquement possible si le suivi des variables a été activé avec la directive générale [7.1.1.27 ini.track-vars](#) ou avec la directive locale `<? php_track_vars ?>`.

`HTTP_GET_VARS`

- Un tableau associatif des variables passées au script courant via les HTTP GET. Uniquement possible si le suivi des variables a été activé avec la directive générale [7.1.1.27 ini.track-vars](#) ou avec la directive locale `<? php_track_vars ?>`.

`HTTP_POST_VARS`

- Un tableau associatif des variables passées au script courant via les HTTP POST. Uniquement possible si le suivi des variables a été activé avec la directive générale [7.1.1.27 ini.track-vars](#) ou avec la directive locale `<? php_track_vars ?>`.

`HTTP_POST_FILES`

- Un tableau associatif contenant les informations sur les fichiers téléchargés avec la méthode HTTP POST. Reportez vous au chapitre [8.4.1 Chargements de fichiers par méthode POST](#) pour plus de détails sur le contenu de `$HTTP_POST_FILES`.

\$HTTP_POST_FILES n'est disponible que dans PHP 4.0.0 et plus récent.

HTTP_ENV_VARS

- Un tableau associatif des variables passées au script par l'environnement parent.

HTTP_SERVER_VARS

- Un tableau associatif des variables passées au script par le serveur HTTP. Ces variables sont analogues aux variables décrites ci-dessus.

9.9.3 Portée des variables

[\[Notes en ligne\]](#)

La portée d'une variable dépend du contexte dans lequel la variable est définie. Pour la plupart des variables, la portée concerne la totalité d'un script PHP. Mais, lorsque vous définissez une fonction, la portée d'une variable définie dans cette fonction est locale à la fonction. Par exemple:

```
<?php
$a = 1;
include "b.inc";
```

Ici, la variable `$a` sera accessible dans le script inclus ``b.inc'`. Cependant, dans les fonctions définies par l'utilisateur, une nouvelle définition de cette variable sera donnée, limitée à la fonction. Toute variable utilisée dans une fonction est par définition, locale. Par exemple :

```
<?php
$a = 1; /* portée global */
Function Test () {
echo $a; /* portée locale */
}
Test ();
```

Le script n'affichera rien à l'écran car la fonction [echo\(\)](#) utilise la variable locale `$a`, et celle-ci n'a pas été assignée préalablement dans la fonction. Vous pouvez noter que ce concept diffère un petit peu du langage C dans lequel une variable globale est automatiquement accessible dans les fonctions, à moins d'être redéfinie localement dans la fonction. Cela peut poser des problèmes si vous redéfinissez des variables globales localement. En PHP, une variable globale doit être déclarée à l'intérieure de chaque fonction afin de pouvoir être utilisée dans cette fonction. Par exemple:

```
<?php
$a = 1;
$b = 2;
Function somme() {
global $a, $b;
    $b = $a + $b;
}
somme();
echo $b;
```

Le script ci-dessus va afficher la valeur "3". En déclarant global les variables `$a` et `$b` localement dans la fonction, toutes les références à ces variables concerneront les variables globales. Il n'y a aucune limite au nombre de variables globales qui peuvent être manipulées par une fonction.

Une deuxième méthode pour accéder aux variables globales est d'utiliser le tableau associatif prédéfini `$GLOBALS`. Le précédent exemple peut être réécrit de la manière suivante:

```
<?php
$a = 1;
$b = 2;
function somme() {
    $GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];
}
somme();
echo $b;
```

Le tableau `$GLOBALS` est un tableau associatif avec le nom des variables globales comme clef et les valeurs des éléments du tableau comme valeur des variables.

Une autre caractéristique importante de la portée des variables est la notion de variable *static*. Une variable statique a une portée locale uniquement mais elle ne perd pas sa valeur lorsque le script appelle la fonction. Prenons l'exemple suivant:

```
<?php
function test() {
    $a = 0;
    echo $a;
    $a++;
}
?>
```

Cette fonction est un peu inutile car à chaque fois qu'elle est appelée, elle initialise `$a` à 0 et affiche "0". L'incrémement de la variable (`$a++`) ne sert pas à grand chose car dès que la fonction est terminée la variable disparaît. Pour faire une fonction de comptage utile, c'est-à-dire qui ne perdra pas la trace du compteur, la variable `$a` est déclarée comme une variable statique:

```
<?php
function test() {
    static $a = 0;
    echo $a;
    $a++;
}
?>
```

Maintenant, à chaque fois que la fonction `Test()` est appelée, elle affichera une valeur de `$a` incrémentée de 1. Les variables statiques sont essentielles lorsque vous faites des appels récursifs à une fonction. Une fonction récursive est une fonction qui s'appelle elle-même. Il faut faire attention lorsque vous écrivez une fonction récursive car il est facile de faire une boucle infinie. Vous devez vérifier que vous avez bien une condition qui permet de terminer votre récursivité. La fonction suivante compte récursivement jusqu'à 10:

```
<?php
function test() {
    static $count = 0;
    $count++;
    echo $count;
    if ($count < 10) {
        test();
    }
    $count--;
}
?>
```

9.9.4 Les variables dynamiques

[\[Notes en ligne\]](#)

Il est pratique d'avoir parfois des noms de variables qui sont variables. C'est-à-dire un nom de variable qui affecté et utilisé dynamiquement. Une variable classique est affecté avec l'instruction suivante:

```
<?php
$a = "bonjour";
?>
```

Une variable dynamique prend la valeur d'une variable et l'utilise comme nom d'une autre variable. Dans l'exemple ci-dessous, *hello*, peut être utilisé comme le nom d'une variable en utilisant le "\$\$" précédant la variable. C'est-à-dire

```
<?php
$$a = "monde";
?>
```

A ce niveau, deux variables ont été définies et stockées dans l'arbre des symboles PHP: \$a avec comme valeur "bonjour" et \$bonjour avec comme valeur "monde". Alors, l'instruction

```
<?php
echo "$a ${$a}";
?>
```

produira le même affichage que :

```
<?php
echo "$a $bonjour";
?>
```

c'est-à-dire : *bonjour monde*.

Afin de pouvoir utiliser les variables dynamiques avec les tableaux, vous avez à résoudre un problème ambigu. Si vous écrivez \$\$a[1], le parseur a besoin de savoir si vous parlez de la variable qui a pour nom \$a[1] ou bien si vous voulez l'index [1] de la variable \$\$a. La syntaxe pour résoudre cette ambiguïté est la suivante: \${\$a[1]} pour le premier cas, et \${\${\$a}}[1] pour le deuxième.

9.9.5 Variables externes à PHP

[\[Notes en ligne\]](#)

9.9.5.1 Formulaires HTML (GET et POST)

[\[Notes en ligne\]](#)

Lorsqu'un formulaire est envoyé à un script PHP, toutes les variables du formulaire seront automatiquement disponibles dans le script. Par exemple, considérons le formulaire suivant:

Exemple avec un formulaire simple

```
<form action="foo.php3" method="post">
Name: <input type="text" name="name"><br>
      <input type="submit">
</form>
```

Lorsque ce formulaire est envoyé, le PHP va créer la variable `$name`, qui contiendra la valeur que vous avez entré dans le champs *Name*: du formulaire.

Le PHP permet aussi l'utilisation des tableaux dans le contexte de formulaire, mais seulement des tableaux à une seule dimension. Comme cela, vous pouvez rassembler des variables ou utiliser cette fonctionnalité pour récupérer les valeurs d'un choix multiple :

Variables complexes de formulaire

```
<form action="array.php" method="post">
Name: <input type="text" name="personal[name]"><br>
Email: <input type="text" name="personal[email]"><br>
Beer: <br>
      <select multiple name="vin[]">
        <option value="medoc">Médoc
        <option value="chablis">Chablis
        <option value="riesling">Riesling
      </select>
      <input type="submit">
</form>
```

Si l'option "track_vars" est activée, soit par l'option de compilation [7.1.1.27 ini.track-~~vars~~](#), soit par la directive de configuration `<? php_track_vars ?>`, les variables transmises par les méthodes POST et GET pourront aussi être trouvées dans le tableau associatif global `$HTTP_POST_VARS` ou `$HTTP_GET_VARS` suivant la méthode utilisée.

9.9.5.1. Bouton "submit" sous forme d'image

[\[Notes en ligne\]](#)

Lorsque vous envoyez le résultat d'un formulaire, vous pouvez utiliser une image au lieu du bouton "submit" standard en utilisant un tag :

```
<input type="image" src="image.gif" name="sub">
```

Lorsqu'un utilisateur clique sur l'image, le formulaire sera transmis au serveur avec deux variables de plus, `sub_x` et `sub_y`. Ces deux variables contiennent les coordonnées de l'endroit où l'utilisateur a cliqué. Les utilisateurs expérimentés remarqueront que les noms de variables sont transmis avec une virgule à la place du caractère "_", mais le PHP fait la conversion automatiquement.

9.9.5.2 HTTP Cookies

[\[Notes en ligne\]](#)

Le PHP supporte les cookies HTTP de manière totalement transparente, comme défini dans les [Netscape's Spec](#). Les cookies sont un mécanisme permettant de stocker des données sur la machine cliente à des fins d'authentification de l'utilisateur. Vous pouvez établir un cookie grâce à la fonction [setcookie\(\)](#). Les cookies font partie intégrante du "header" HTTP, et donc la fonction [setcookie\(\)](#) doit être appelé avant que le moindre affichage ne soit envoyé au navigateur. C'est la même restriction que pour la fonction [header\(\)](#). Tout cookie envoyé depuis le client sur le serveur sera automatiquement stocké sous forme de variable, comme pour la méthode POST ou GET.

Si vous souhaitez assigner plusieurs valeurs à un seul cookie, il vous faut ajouter les caractères `[]` au nom de votre cookie. Par exemple :


```
<?php
setcookie ("MonCookie[]", "test", time()+3600);
?>
```

Il est à noter qu'un cookie remplace le cookie précédent par un cookie de même nom tant que le "path" ou le domaine sont identiques. Donc, pour une application de caddie, vous devez implémenter un compteur et l'incrémenter au fur et à mesure. C'est-à-dire:

Exemple avec setcookie()

```
<?php
$compte++;
SetCookie ("Compte", $compte, time()+3600);
SetCookie ("Caddie[$compte]", $item, time()+3600);
?>
```

9.9.5.3 Variables d'environnement

[\[Notes en ligne\]](#)

Le PHP fait en sorte que les variables d'environnement soient accessibles directement comme des variables PHP normales.

```
<?php
echo $HOME; /* Affiche la valeur de la variable d'environnement HOME,
si celle-ci est affectée. */
?>
```

Même si le PHP crée les variables lors de l'utilisation des méthodes GET, POST et cookie, il est de temps en temps préférable de transmettre explicitement la valeur de la variable afin d'être sûr de la valeur. La fonction [getenv\(\)](#) peut être utilisé pour récupérer la valeur des variables d'environnement. Vous pouvez aussi affecter une variable d'environnement grâce à la fonction [putenv\(\)](#).

9.9.5.4 Cas des points dans les noms de variables

[\[Notes en ligne\]](#)

Typiquement, PHP ne modifie pas les noms des variables lorsqu'elles sont passées à un script. Cependant, il faut noter que les points (.) ne sont pas autorisés dans les noms de variables PHP. Pour cette raison, jetez un oeil sur :

```
<?php
$varname.ext; /* nom de variable invalide */
?>
```

Dans ce cas, l'analyseur croit voir la variable nommée \$varname, suivie par l'opérateur de concaténation, et suivi encore par la chaîne non-guillemetée (une chaîne sans guillemets, et qui n'a pas de signification particulière). Visiblement, ce n'est pas ce qu'on attendait...

Pour cette raison, il est important de noter que PHP remplacera automatiquement les points des noms de variables entrantes par des soulignés (underscore).

[9.9.5.5 Détermination du type des variables](#)

[\[Notes en ligne\]](#)

Parce que le PHP détermine le type des variables et les convertit (généralement) comme il faut, ce n'est pas toujours le type de variable que vous souhaitez. PHP inclut des fonctions permettant de déterminer le type d'une variable. Les fonctions [gettype\(\)](#), [is_long\(\)](#), [is_double\(\)](#), [is_string\(\)](#), [is_array\(\)](#), et [is_object\(\)](#).

[9.10 Les références](#)

[\[Notes en ligne\]](#)

[9.10.1 Qu'est ce qu'une référence?](#)

[\[Notes en ligne\]](#)

En PHP, les références sont destinées à appeler le contenu d'une variable avec un autre nom. Ce n'est pas comme en C : ici, les références sont des alias dans la table des symboles. Le nom de la variable et son contenu ont des noms différents, ce qui fait que l'on peut donner plusieurs noms au même contenu. On peut faire l'analogie avec les fichiers sous Unix, et leur nom de fichier : les noms des variables sont les entrées dans un répertoire, tandis que le contenu de la variable est le contenu même du fichier. Faire des références en PHP revient alors à faire des liens sous Unix.

[9.10.2 Que font les références](#)

[\[Notes en ligne\]](#)

Les références vous permettent de faire pointer deux variables sur le même contenu. Par exemple, lorsque vous faites :

```
<?php
$a =& $b
?>
```

cela signifie que ***\$a*** et ***\$b*** pointent sur la même variable. Note : ***\$a*** et ***\$b*** sont complètement égales ici : ce n'est pas ***\$a*** qui pointe sur ***\$b***, ou vice versa. C'est bien ***\$a*** et ***\$b*** qui pointent sur le même contenu.

La même syntaxe peut être utilisée avec les fonctions qui retournent des références, et avec l'opérateur new (PHP 4.0.4 et plus récent):

```
<?php
$bar =& new fooclass();
$foo =& find_var ($bar);
?>
```

Note : A moins d'utiliser la syntaxe ci-dessus, le résultat de `$bar = new fooclass()` ne sera pas la même variable que `$this` dans le constructeur, ce qui signifie que si vous avez utilisé la référence `$this` dans le constructeur, vous devez assigner la référence, ou bien obtenir deux objets différents.

Le deuxième intérêt des références est de pouvoir passer des variables par référence. On réalise ceci en faisant pointer des variables locales vers le contenu des variables de fonction. Exemple :

```
<?php
function foo (&$var) {
    $var++;
}
```

```
$a=5;
foo ($a);
?>
```

\$a vaut 6. Cela provient du fait que dans la fonction *foo*, la variable *\$var* pointe sur le même contenu que *\$a*. Voir aussi les explications détaillées dans [9.10.4 Passage par référence](#). Le troisième intérêt des références est de [9.10.5 Retourner des références](#).

9.10.3 Ce que les références ne sont pas

[\[Notes en ligne\]](#)

Comme précisé ci dessus, les références ne sont pas des pointeurs. Cela signifie que le script suivant ne fera pas de à quoi on peut s'attendre :

```
<?php
function foo (&$var) {
    $var =& $GLOBALS["baz"];
}
foo($bar);
?>
```

Il va se passer que *\$var* dans *foo* sera lié à *\$bar*, mais il sera aussi relié à *\$GLOBALS["baz"]*. Il n'y a pas moyen de le lier *\$bar* à quelque chose d'autre en utilisant le mécanisme de référence, car *\$bar* n'est pas accessible dans la fonction *foo*. (il est représenté par *\$var*, mais *\$var* possède la même valeur, mais n'est pas relié par la table des symboles).

9.10.4 Passage par référence

[\[Notes en ligne\]](#)

Vous pouvez passer des variables par référence, de manière à ce que la fonction modifie ses arguments. La syntaxe est la suivante :

```
<?php
function foo (&$var) {
    $var++;
}
$a=5;
foo ($a);
// $a vaut 6 maintenant
?>
```

Notez qu'il n'y a pas de signe de référence dans l'appel de la fonction, uniquement sur sa définition. La définition de la fonction est suffisante pour passer correctement des arguments par référence.

Les objets suivants peuvent être passés par référence :

- Une variable, i.e. `foo($a)`
- Un nouvel objet, i.e. `foo(new foobar())`
- Une référence, retournée par une fonction :

```
function &bar()
{
    $a = 5;
    return $a;
}
```

```
foo(bar());
```

Voir aussi des détails dans [9.10.5 Retourner des références](#).

Toutes les autres expressions ne doivent pas être passées par référence, car le résultat sera indéfini. Par exemple, les passages par référence suivants sont invalides :

```
function bar() // Notez l'absence de &
{
    $a = 5;
    return $a;
}
foo(bar());
foo($a = 5) // Expression, pas une variable
foo(5) // Constante, pas une variable
```

Ces fonctionnalités sont valables à partir de PHP 4.0.4.

9.10.5 Retourner des références

[\[Notes en ligne\]](#)

Retourner des références est toujours utile lorsque vous voulez utiliser une fonction pour savoir à quoi est liée une variable. Lorsque vous retournez une variable par paramètre, utilisez le code suivant

```
function &find_var ($param) {
    ...code...
    return $found_var;
}
$foo =& find_var ($bar);
$foo->x = 2;
```

Dans cet exemple, la propriété de l'objet est retourné dans *find_var* et lui sera affecté, et non pas à la copie, comme cela sera le cas avec une syntaxe par référence.

Note : Contrairement au passage de paramètre, vous devez utiliser & aux deux endroits, à la fois pour indiquer que vous retournez par référence (pas une copie habituelle), et pour indiquer que vous assigner aussi par référence (pas la copie habituelle).

9.10.6 Détruire une références

[\[Notes en ligne\]](#)

Lorsque vous détruire une référence, vous ne faites que casser le lien entre le nom de la variable et son contenu. Cela ne signifie pas que le contenu est détruit. Par exemple,

```
<?php
$a = 1;
$b =& $a;
unset ($a);
?>
```

Cet exemple ne détruira pas *\$b*, mais juste *\$a*.

Encore une fois, on peut comparer cette action avec la fonction unlink d'Unix.

[9.10.7 Repérer une référence](#)

[\[Notes en ligne\]](#)

De nombreuses syntaxes de PHP sont implementées via le mécanisme de référence, et tout ce qui a été vu concernant les liaisons entre variables s'applique à ces syntaxes. Par exemple, le passage et le retour d'arguments par référence. Quelques autres exemples de syntaxes :

[9.10.7.1 Références global](#)

[\[Notes en ligne\]](#)

Lorsque vous déclarez une variable `global $var`, vous créez en fait, une référence sur une variable globale. Ce qui signifie que

```
<?php
$var =& $GLOBALS[ "var" ];
?>
```

Et que, si vous détruisez la variable *\$var*, la variable globale ne sera pas détruite.

[9.10.7.2 \\$this](#)

[\[Notes en ligne\]](#)

Dans une méthode d'objet, *\$this* est toujours une référence sur l'objet courant.

10 Fonctions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

10.1 Apache

[\[Notes en ligne\]](#)

10.1.1 apache_lookup_uri

[\[Notes en ligne\]](#) [\[Exemples\]](#)

class [apache_lookup_uri](#) (string *filename*)
[PHP 3>= 3.0.4, PHP 4]

[apache_lookup_uri\(\)](#) effectue une requête partielle pour l'URI spécifiée. Cette requête permet de récupérer toutes les informations importantes à propos de la ressource concernée. Les propriétés de la classe renvoyée sont les suivantes :

- status
- the_request
- status_line
- method
- content_type
- handler
- uri
- filename
- path_info
- args
- boundary
- no_cache
- no_local_copy
- allowed
- send_bodyct
- bytes_sent
- byterange
- clength
- unparsed_uri
- mtime
- request_time

Note : *Note: apache_lookup_uri ne fonctionne que lorsque le PHP est installé sous la forme d'un module Apache.*

10.1.2 apache_note

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [apache_note](#) (string *note_name*, string *note_value*)
[PHP 3>= 3.0.2, PHP 4]

[apache_note\(\)](#) est une fonction spécifique au serveur Apache. Cette fonction affecte ou renvoie la valeur de la variable contenue dans la table notes d'Apache. Si la fonction est appelée avec un argument, elle renvoie la valeur courante de la variable `note_name`. Si la fonction est appelée avec deux arguments, la variable `note_name` la valeur `note_value` et la fonction retournera la valeur précédente de la variable `note_name`.

10.1.3 [getallheaders](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getallheaders](#) (void)
[PHP 3, PHP 4]

[getallheaders\(\)](#) renvoie un tableau associatif de tous les entêtes HTTP correspondants à la requête courante.
Note : Note: Vous pouvez récupérer la valeur d'une variable d'une CGI en la lisant à partir des variables d'environnement, ce qui fonctionne aussi bien dans le cas d'une installation en module ou en CGI. Utilisez la fonction [phpinfo\(\)](#) pour avoir une liste de toutes les variables d'environnement disponibles.

Exemple d'utilisation de la fonction [getallheaders\(\)](#)

```
<?php
$headers = getallheaders();
while (list($entete, $valeur) = each($headers)) {
    echo "$entete: $valeur<br>\n";
}
?>
```

Cette exemple est un exemple d'affichage de toutes les entêtes de la requête courante. *Note : Note: La fonction [getallheaders\(\)](#) ne fonctionne que si PHP est installé comme module **Apache**.*

10.1.4 [virtual](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [virtual](#) (string *filename*)
[PHP 3, PHP 4]

[virtual\(\)](#) est une fonction spécifique au serveur Apache. Elle est équivalente à la directive "`<!--#include virtual...-->`" lorsque vous utilisez le module include d'Apache. Cette fonction effectue une sous-requête Apache. C'est très utile lorsque vous utilisez des scripts CGI, des fichiers .shtml ou n'importe quel type de fichier qui doit être analysé par le serveur Apache. Il est à noter que lors de l'utilisation avec des scripts CGI, ces derniers doivent générer une entête valide, c'est-à-dire, au minimum une entête "Content-Type". Pour les fichiers PHP, il est conseillé d'utiliser les fonctions [include\(\)](#) et [require\(\)](#). [virtual\(\)](#) ne peut pas être utilisé pour inclure un fichier qui est lui même un fichier PHP.

10.1.5 [ascii2ebcdic](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ascii2ebcdic](#) (string *ascii_str*)
[PHP 3>= 3.0.17]

[ascii2ebcdic\(\)](#) est une fonction spécifique à Apache, qui n'est disponible que sur les OS qui gère le format

EBCDIC (OS/390, BS2000). Elle traduit la chaîne ASCII *ascii_str* en son équivalent EBCDIC (avec protection des données binaires) et retourne le résultat.

Voir aussi la fonction complémentaire : [ebcdic2ascii\(\)](#)

10.1.6 ebcdic2ascii

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ebcdic2ascii](#)(string *ebcdic_str*)

[PHP 3>= 3.0.17]

[ebcdic2ascii\(\)](#) est une fonction spécifique à Apache, qui n'est disponible que sur les OS qui gèrent le format EBCDIC (OS/390, BS2000). Elle traduit la chaîne EBCDIC *ebcdic_str* en son équivalent ASCII (avec protection des données binaires) et retourne le résultat.

Voir aussi la fonction complémentaire : [ascii2ebcdic\(\)](#)

10.2 Tableaux

[\[Notes en ligne\]](#)

Ces fonctions vous permettent de manipuler et de traiter les tableaux de nombreuses façons. Les tableaux sont très efficaces dès qu'il s'agit de stocker, gérer et traiter des données en groupe.

Les tableaux simples et multi-dimensionnels sont supportés et peuvent être créés par l'utilisateur, ou par une fonction. Il y a des fonctions spécifiques qui remplissent des tableaux à partir de résultats de requêtes, et de nombreuses fonctions retournent un tableau.

Voir aussi [is_array\(\)](#), [explode\(\)](#), [implode\(\)](#), [split\(\)](#) et [join\(\)](#).

10.2.1 array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array](#)(mixed ...)

[array\(\)](#) retourne un tableau créé avec les paramètres passés. On peut attribuer un index particulier à une valeur avec l'opérateur `=>`.

Note : [array\(\)](#) est un élément de langage utilisé pour représenter des tableaux littéraux, et non pas une fonction au sens strict du terme.

La syntaxe "index => valeur", séparés par des virgules, définit les index et leur valeur. Un index peut être une chaîne ou un nombre. Si l'index est omis, un index numérique sera automatiquement généré (commençant à 0). Si l'index est un entier, le prochain index généré prendra la valeur d'index la plus grande + 1. Notez que si deux index identiques sont définis, le dernier remplacera le premier.

L'exemple suivant montre comment créer un tableau à deux dimensions, comment spécifier les index d'un tableau associatif, et comment générer automatiquement des index numériques.

Exemple avec array()

```
<?php
$fruits = array (
    "fruits" => array ("a" => "orange", "b" => "banane", "c" => "pomme"),
    "nombres" => array (1, 2, 3, 4, 5, 6),
    "trous"    => array ("premier", 5 => "deuxième", "troisième")
);
```



```
?>
```

Index automatique d'un tableau avec array()

```
<?php
$array = array( 1, 1, 1, 1, 1, 8=>1, 4=>1, 19, 3=>13);
print_r($array);
?>
```

qui affichera :

```
Array
(
    [0] => 1
    [1] => 1
    [2] => 1
    [3] => 13
    [4] => 1
    [8] => 1
    [9] => 19
)
```

Notez bien que l'index '3' est défini deux fois, et conserve finalement sa dernière valeur de 13. L'index '4' est défini après l'index '8', et l'index généré suivant (valeur 19) est 9, puisque le plus grand index est alors 8. Cet exemple crée un tableau dont les index commencent à 1.

Tableau d'index commençant à 1

```
<?php
$firstquarter = array(1 => 'Janvier', 'Février', 'Mars');
print_r($firstquarter);
?>
```

qui affichera :

```
Array
(
    [1] => 'Janvier'
    [2] => 'Février'
    [3] => 'Mars'
)
```

Voir aussi : [list\(\)](#).

10.2.2 array_count_values

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_count_values](#) (array *input*)
[PHP 4 >= 4.0b4]

[array_count_values\(\)](#) retourne un tableau contenant les valeurs du tableau *input* comme clés et leurs fréquence comme valeur.

Exemple avec `array_count_values()`

```
<?php
$array = array(1, "bonjour", 1, "monde", "bonjour");
array_count_values($array); // retourne array(1=>2, "bonjour"=>2, "monde"=>1)
?>
```

Note : [`array_count_values\(\)`](#) a été ajoutée dans PHP 4.0.

10.2.3 array_diff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [`array_diff\(\)`](#)(array *array1*, array *array2*, array ...)

[PHP 4 >= 4.0.1]

[`array_diff\(\)`](#) retourne un tableau qui contient toutes les valeurs du tableau *array1* qui sont absentes de tous les autres arguments. Notez que les clés sont préservées.

Exemple avec `array_diff()`

```
<?php
$array1 = array ("a" => "vert", "rouge", "bleu");
$array2 = array ("b" => "vert", "jaune", "rouge");
$result = array_diff ($array1, $array2);
?>
```

\$result contient array("bleu");

Voir aussi [`array_intersect\(\)`](#).

10.2.4 array_flip

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [`array_flip\(\)`](#)(array *trans*)

[PHP 4 >= 4.0b4]

[`array_flip\(\)`](#) retourne un tableau dont les clés sont les valeurs du précédent tableau, et les valeurs sont les clés.

Exemple avec `array_flip()`

```
<?php
$trans = array_flip ($strans);
$original = strtr ($str, $trans);
?>
```

Note : [`array_flip\(\)`](#) a été ajoutée dans PHP 4.0.

10.2.5 [array_intersect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_intersect](#) (array *array1*, array *array2*, array ...)

[PHP 4 >= 4.0.1]

[array_intersect\(\)](#) retourne un tableau contenant toutes les valeurs de *array1* qui sont présentes dans tous les autres arguments. Notez que les clés sont préservées.

Exemple avec array_intersect()

```
<?php
$array1 = array ("a" => "vert", "rouge", "bleu");
$array2 = array ("b" => "vert", "jaune", "rouge");
$result = array_intersect ($array1, $array2);
?>
```

\$result contient array ("a" => "vert", "rouge");.

Voir aussi [array_diff\(\)](#).

10.2.6 [array_keys](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_keys](#) (array *input*, mixed *search_value*)

[PHP 4]

[array_keys\(\)](#) retourne les clés numériques et littérales du tableau *input*.

Si l'option *search_value* est spécifiée, seules les clés ayant cette valeur seront retournées. Sinon, toutes les clés de *input* sont retournées.

Exemple avec array_keys()

```
<?php
$array = array(0 => 100, "couleur" => "rouge");
array_keys($array);
// retourne array(0, "couleur")
$array = array("bleu", "rouge", "vert", "bleu", "bleu");
array_keys($array, "bleu");
// retourne array(0, 3, 4)
$array = array( "couleur" => array("bleu", "rouge", "vert"),
               "taille"   => array("petit", "moyen", "grand") );
array_keys($array);
// retourne array("couleur", "taille")
?>
```

Note : [array_keys\(\)](#) a été ajoutée dans PHP 4. Ci-dessous, voici une implémentation qui fonctionnera sous PHP 3:

Implémentation de array_keys() pour les utilisateurs de php 3

```
<?php
function array_keys ($arr, $term="") {
```

```

    $t = array();
    while (list($k,$v) = each($arr)) {
        if ($term && $v != $term)
            continue;

        $t[] = $k;

    }
    return $t;
}
?>

```

} Voir aussi [array_values\(\)](#).

10.2.7 array_merge

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_merge](#) (array *array1*, array *array2*, ...)

[PHP 4]

[array_merge\(\)](#) rassemble les éléments de plusieurs tableaux ensemble, en ajoutant les valeurs de l'un à la fin de l'autre. Le résultat est un tableau.

Si les tableaux ont des clés en commun, la dernière valeur rencontrée écrasera l'ancienne. Pour les valeurs numériques, cela n'arrive pas, car alors, les valeurs sont ajoutées en fin de tableau.

Exemple avec array_merge()

```

<?php
$array1 = array ("couleur" => "rouge", 2, 4);
$array2 = array ("a", "b", "couleur" => "vert", "forme" => "trapézoïde");
array_merge ($array1, $array2);
?>

```

Le résultat sera array("couleur" => "vert", 2, 4, "a", "b", "forme" => "trapézoïde").

Note : [array_merge\(\)](#) a été ajoutée dans PHP 4.0.

10.2.8 array_merge_recursive

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_merge_recursive](#) (array *array1*, array *array2*, array ...)

[PHP 4 >= 4.0.1]

[array_merge_recursive\(\)](#) rassemble tous les éléments de plusieurs tableaux ensemble, en ajoutant les éléments de l'un à la suite des éléments du précédent. [array_merge_recursive\(\)](#) retourne le tableau résultant.

Si les tableaux passés en arguments ont les mêmes clés (chaînes de caractères), les valeurs sont alors rassemblées dans un tableau, de manière récursive, de façon à ce que, si l'une de ces valeurs est un tableau elle-même, la fonction la rassemblera avec les valeurs de l'entrée courante. Cependant, si deux tableaux ont la même clé numérique, la dernière valeur n'écrasera pas la précédente, mais sera ajouté à la fin du tableau.

Exemple avec array_merge_recursive()

```

<?php

```

```
$ar1 = array ("couleur" => array ("favorie" ?> "rouge"), 5);
$ar2 = array (10, "couleur" ?> array ("favorie" ?> "vert", "rouge"));
$result = array_merge_recursive ($ar1, $ar2);
?>
```

Le résultat sera array("couleur" => array("favorie" => array("rouge", "vert"), "bleu"), 5, 10).
Voir aussi [array_merge\(\)](#).

10.2.9 [array_multisort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [array_multisort](#) (array *ar1*, mixed *arg* , mixed ... , array ...)
[PHP 4 >= 4.0b4]

[array_multisort\(\)](#) sert à trier simultanément plusieurs tableaux, ou bien à trier un tableau multi-dimensionnel, suivant l'une ou l'autre de ses dimensions. Les clés sont préservées.

Les tableaux passés en arguments sont traités comme les colonnes d'une table, triées par lignes (un peu comme la clause SQL ORDER BY). Le premier tableau est la clé primaire de tri. Les valeurs du premier tableaux qui sont égales, sont triées grâce au tableau suivant, et ainsi de suite...

La structure des arguments de [array_multisort\(\)](#) est un peu inhabituelle, mais elle est plus souple. Le premier argument DOIT être un tableau, mais les arguments suivants peuvent être des tableaux ou une ou deux options de tri, prises dans les valeurs suivantes :

Options de tri :

- SORT_ASC – Tri en ordre ascendant
- SORT_DESC – Tri en ordre descendant

Options de type de tri:

- SORT_REGULAR – Comparaison normale des valeurs
- SORT_NUMERIC – Comparaison numérique des valeurs
- SORT_STRING – Comparaison alphabétique des valeurs

Une seule option de tri de chaque type peut être appliquée après un tableau. Une option ne s'applique qu'au tableau précédent. Tous les autres sont mis par défaut à SORT_ASC et SORT_REGULAR.
Retourne TRUE en cas de succès, FALSE sinon.

Trier plusieurs tableaux

```
<?php
$ar1 = array ("10", 100, 100, "a");
$ar2 = array (1, 3, "2", 1);
array_multisort ($ar1, $ar2);
?>
```

Dans cet exemple, Dans cet exemple, après le tri, le premier tableau contient 10, "a", 100, 100; Le deuxième tableau contient 1, 1, 2, "3". Les entrées du second tableau correspondent aux valeurs jumelles du premier tableau (100 et 100), sont aussi triées.

Classer un tableau multidimensionnel

```
<?php
$ar = array (array ("10", 100, 100, "a"), array (1, 3, "2", 1));
array_multisort ($ar[0], SORT_ASC, SORT_STRING,
                 $ar[1], SORT_NUMERIC, SORT_DESC);
?>
```

Dans cet exemple, après le tri, le premier tableau contient 10, 100, 100, "a" (tri alphabétique, ordre croissant); Le deuxième tableau contient 1, 3, "2", 1 (tri numérique, ordre décroissant).

10.2.10 array_pad

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_pad](#)(array *input*, int *pad_size*, mixed *pad_value*)
[PHP 4 >= 4.0b4]

[array_pad\(\)](#) retourne une copie du tableau *input* complété jusqu'à la taille de *pad_size* avec la valeur *pad_value*. si *pad_size* est positif alors le tableau est complété à droite, si il est négatif, il est complété à gauche. Si la valeur absolue de *pad_size* est plus petite que la taille du tableau *input* alors le tableau n'est pas complété.

Exemple avec array_pad()

```
<?php
$input = array (12, 10, 9);
$result = array_pad ($input, 5, 0);
// Le résultat est array (12, 10, 9, 0, 0)
$result = array_pad ($input, -7, -1);
// Le résultat est array (-1, -1, -1, -1, 12, 10, 9)
$result = array_pad ($input, 2, "noop");
// pas complété
?>
```

10.2.11 array_pop

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [array_pop](#)(array *array*)
[PHP 4]

[array_pop\(\)](#) dépile et retourne le dernier élément du tableau *array*, le raccourcissant d'un élément.

Exemple avec array_pop()

```
<?php
$stack = array ("orange", "pomme", "framboise");
$fruit = array_pop ($stack);
?>
```

Après ceci, \$stack n'a plus que 2 éléments: "orange" et "pomme", tandis que \$fruit contient "framboise".

Voir aussi [array_push\(\)](#), [array_shift\(\)](#), et [array_unshift\(\)](#). Note : [array_pop\(\)](#) a été ajoutée dans PHP 4.0.

10.2.12 array_push

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [array_push](#) (array *array*, mixed *var*, ...)
[PHP 4]

[array_push\(\)](#) considère *array* comme une pile, et empile les variables passées en paramètres à la fin de *array*. La longueur du tableau *array* augmente d'autant. Cela a le même effet que :

```
<?php
$array[] = $var;
?>
```

repeté pour chaque *var*.

Retourne le nouveau nombre d'éléments du tableau.

Exemple avec array_push()

```
<?php
$stack = array (1, 2);
array_push($stack, "+", 3);
?>
```

Cet exemple fait que \$stack a 4 éléments: 1, 2, "+", et 3.

Voir aussi: [array_pop\(\)](#), [array_shift\(\)](#), et [array_unshift\(\)](#). Note : [array_push\(\)](#) a été ajoutée dans PHP 4.0.

10.2.13 array_reverse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_reverse](#) (array *array*, bool *preserve_keys*)
[PHP 4 >= 4.0b4]

[array_reverse\(\)](#) prend le tableau *array* et retourne un nouveau tableau qui contient les mêmes éléments mais dans l'ordre inverse, en préservant les clés si le paramètre *preserve_keys* vaut TRUE.

Exemple avec array_reverse()

```
<?php
$input = array ("php", 4.0, array ("rouge", "vert"));
$result = array_reverse ($input, TRUE);
?>
```

Au final, \$result contient (array ("rouge", "vert"), 4.0, "php"). Mais *\$result2[0]* vaut toujours "php". Note : [array_reverse\(\)](#) a été ajoutée en PHP 4.0 Beta 3.

Note : Le second paramètre *preserve_keys* a été ajouté en PHP 4.0.3.

10.2.14 array_rand

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [array_rand](#)(array *input*, int *num_req*)

[PHP 4 >= 4.0.0]

[array_rand\(\)](#) est pratique lorsque vous voulez sélectionner une ou plusieurs valeurs au hasard dans un tableau. Le paramètre *input* est un tableau, et *num_req* spécifie le nombre de valeurs que vous voulez obtenir (par défaut, c'est 1).

Si vous ne demandez qu'une entrée, [array_rand\(\)](#) retourne l'index de la valeur. Sinon, elle retourne un tableau d'index. Cela vous permet de faire une sélection au hasard de clés, ou bien de valeur.

N'oubliez pas d'appeler [srand\(\)](#) pour initialiser le générateur de nombres aléatoires.

Exemple avec array_rand()

```
<?php
srand ((double) microtime() * 10000000);
$input = array ("Neo", "Morpheus", "Trinitée", "Cypher", "Tank");
$rand_keys = array_rand ($input, 2);
print $input[$rand_keys[0]]."\n";
print $input[$rand_keys[1]]."\n";
?>
```

10.2.15 array_shift

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [array_shift](#)(array *array*)

[PHP 4]

[array_shift\(\)](#) extrait la première valeur d'un tableau et la retourne, en raccourcissant le tableau d'un élément, et en déplaçant tous les éléments vers le bas.

Exemple avec array_shift()

```
<?php
$args = array ("-v", "-f");
$opt = array_shift ($args);
?>
```

Cet exemple aura pour résultat que *\$args* ne contiendra plus que "-f", et *\$opt* contient "-v".

Voir aussi [array_unshift\(\)](#), [array_push\(\)](#), et [array_pop\(\)](#). Note : [array_shift\(\)](#) a été ajoutée dans PHP 4.0.

10.2.16 array_slice

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_slice](#)(array *array*, int *offset*, int *length*)
[PHP 4]

[array_slice\(\)](#) retourne une série d'éléments du tableau *array* commençant à l'offset *offset* et représentant *length* éléments.

Si *offset* est positif, la série commencera à cet offset dans le tableau *array*. Si *offset* est négatif, cette série commencera à l'offset *offset* mais en commençant à la fin du tableau *array*.

Si *length* est donné et positif, alors la série aura autant d'éléments. Si *length* est donné et négatif, les éléments seront pris dans l'ordre inverse. Si *length* est omis, la séquence lira tous les éléments du tableau, depuis l'*offset* précisé jusqu'à la fin du tableau.

Exemple avec array_slice()

```
<?php
$input = array ("a", "b", "c", "d", "e");
$output = array_slice ($input, 2);      // retourne "c", "d", et "e"
$output = array_slice ($input, 2, -1);  // retourne "c", "d"
$output = array_slice ($input, -2, 1);  // retourne "d"
$output = array_slice ($input, 0, 3);   // retourne "a", "b", et "c"
?>
```

Voir aussi [array_splice\(\)](#). Note : [array_slice\(\)](#) a été ajoutée dans PHP 4.0.

10.2.17 array_splice

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_splice](#)(array *input*, int *offset*, int *length* , array *replacement*)
[PHP 4]

[array_splice\(\)](#) supprime les éléments désignés par *offset* et *length* du tableau *input* et les remplace par les éléments du tableau *replacement*, si ce dernier est présent.

Si *offset* est positif, la série commencera à cet offset dans le tableau *input*. Si *offset* est négatif, cette série commencera à l'offset *offset* mais en commençant à la fin du tableau *input*.

Si *length* est donné et positif, alors la série aura autant d'éléments. Si *length* est donné et négatif, les éléments seront pris dans l'ordre inverse. Si *length* est omis, la séquence lira tous les éléments du tableau, depuis l'offset précisé jusqu'à la fin du tableau. Conseil : pour supprimer tous les éléments du tableau depuis *offset* jusqu'à la fin, même si un tableau de remplacement *replacement* est spécifié, utilisez `count(count($input))` à la place de *length*.

Si *replacement* est précisé, alors les éléments supprimés sont remplacés par les éléments de ce tableau. Si l'*offset* et *length* sont tels que la taille du tableau ne change pas, alors les éléments du tableau de remplacement *replacement* sont insérés à partir de l'offset *offset*.

Conseil : si le tableau de remplacement ne contient qu'un seul élément, il n'est pas obligatoire de forcer le type en tableau avec [array\(\)](#), à moins que cette variable ne soit elle-même un tableau.

Les codes suivants sont équivalents :

```
<?php
array_push($input, $x, $y)      array_splice($input, count($input), 0, array($x, $y))
array_pop($input)              array_splice($input, -1)
array_shift($input)            array_splice($input, 0, 1)
array_unshift($input, $x, $y)  array_splice($input, 0, 0, array($x, $y))
```

```
$a[$x] = $y          array_splice($input, $x, 1, $y)
?>
```

[array_splice\(\)](#) retourne le tableau des éléments supprimés.

Exemples avec array_splice()

```
<?php
$input = array("rouge", "vert", "bleu", "jaune");
array_splice($input, 2);      // $input est array("rouge", "vert")
array_splice($input, 1, -1);  // $input est array("rouge", "yellow")
array_splice($input, 1, count($input), "orange");
                                // $input est array("rouge", "orange")
array_splice($input, -1, 1, array("noir", "marron"));
                                // $input est array("rouge", "vert",
                                //                      "bleu", "noir", "marron")
?>
```

Voir aussi [array_slice\(\)](#). Note : [array_splice\(\)](#) a été ajoutée dans PHP 4.0.

10.2.18 array_unique

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_unique](#) (array *array*)
[PHP 4 >= 4.0.1]

[array_unique\(\)](#) prend le tableau *array* et retourne un nouveau tableau, complètement dédoublonné. Notez que les clés sont préservées.

Exemple avec array_unique()

```
<?php
$input = array ("a" => "vert", "rouge", "b" => "vert", "bleu", "rouge");
$result = array_unique ($input);
?>
```

\$result contient array ("a" => "vert", "rouge", "bleu");.

10.2.19 array_unshift

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [array_unshift](#) (array *array*, mixed *var*, ...)
[PHP 4]

[array_unshift\(\)](#) ajoute les éléments passé en argument au début du tableau *array*. Notez que les éléments sont ajoutés comme un tout, et qu'ils restent dans le même ordre.

Retourne le nouveau nombre d'éléments du tableau *array*.

Exemples avec `array_unshift()`

```
<?php
$queue = array("p1", "p3");
array_unshift($queue, "p4", "p5", "p6");
?>
```

Le résultat de cet exemple est que `$queue` aura 5 éléments, à savoir: "p4", "p5", "p6", "p1", et "p3".
Voir aussi [array_shift\(\)](#), [array_push\(\)](#), et [array_pop\(\)](#). Note : [array_unshift\(\)](#) a été ajoutée dans PHP 4.0.

10.2.20 array_values

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [array_values](#) (array *input*)

[PHP 4]

[array_values\(\)](#) retourne les valeurs du tableau *input*.

Exemples avec `array_values()`

```
<?php
$array = array("taille" => "XL", "couleur" => "or");
array_values($array);    // // retourne array("XL", "or")
?>
```

Note : [array_values\(\)](#) a été ajoutée dans PHP 4. Ci dessous, voici une implémentation pour ceux qui utilisent toujours PHP 3.

Implémentation de `array_values()` pour les utilisateurs php 3

```
<?php
function array_values ($arr) {
    $t = array();
    while (list($k, $v) = each ($arr)) {
        $t[] = $v;
    }
    return $t;
}
?>
```

10.2.21 array_walk

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [array_walk](#) (array *arr*, string *func*, mixed *userdata*)

[PHP 3>= 3.0.3, PHP 4]

[array_walk\(\)](#) exécute la fonction *func* avec chaque élément du tableau *arr*. Les éléments sont passés en tant que premier argument de la fonction *func*;

Si *func* a besoin de plus d'un argument, une alerte sera générée pour chaque appel de *func*. Ces alertes sont

supprimées en ajoutant le suffixe " avant l'appel de [array_walk\(\)](#) ou simplement en utilisant [error_reporting\(\)](#).

Note : Si **func** doit travailler avec les véritables valeur du tableau, spécifiez que le premier paramètre de **func** doit être passé par référence. Alors, les éléments seront directement modifiés dans le tableau.

Note : Passez les clés et userdata à **func** a été ajouté dans PHP 4.0.

Dans PHP 4, [reset\(\)](#) doit être appelés si nécessaire, car [array_walk\(\)](#) ne réinitialise pas automatiquement le tableau.

Exemple avec `array_walk()`

```
<?php
$fruits = array("d"=>"citron", "a"=>"orange", "b"=>"banane", "c"=>"pomme");
function test_alter( $item1 ) {
    $item1 = 'bidon';
}
function test_print( $item2 ) {
    echo "$item2<br>\n";
}
array_walk( $fruits, 'test_print' );
array_walk( $fruits, 'test_alter' );
array_walk( $fruits, 'test_print' );
?>
```

Voir aussi [each\(\)](#) et [list\(\)](#).

10.2.22 [arsort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [arsort](#)(array *array*)

[PHP 3, PHP 4]

[arsort\(\)](#) trie un tableau de telle manière que la corrélation entre les index et les valeurs soient conservées.

L'usage principal est lors de tri de tableaux associatifs où l'ordre des éléments est important.

Exemple avec `arsort()`

```
<?php
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
arsort( $fruits);
for (reset( $fruits); $key = key( $fruits); next( $fruits)) {
    echo "fruits[$key] = ".$fruits[$key]."\n";
}
?>
```

Cet exemple va afficher: fruits[a] = orange fruits[d] = papaye fruits[b] = banane fruits[c] = ananas Les fruits ont été triés en ordre alphabétique inverse, et leur index respectifs ont été conservé.

Voir aussi: [asort\(\)](#), [rsort\(\)](#), [ksort\(\)](#), et [sort\(\)](#).

10.2.23 [asort](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [asort](#)(array *array*)

[PHP 3, PHP 4]

[asort\(\)](#) trie un tableau de telle manière que la corrélation entre les index et les valeurs soient conservées. L'usage principal est lors de tri de tableaux associatifs où l'ordre des éléments est important.

Exemple avec asort()

```
<?php
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
asort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
}
?>
```

Cet exemple va afficher: fruits[c] = ananas fruits[b] = banane fruits[d] = papaye fruits[a] = orange Les fruits ont été triés par ordre alphabétique, et leur index respectifs ont été conservé.

Voir aussi [arsort\(\)](#), [rsort\(\)](#), [ksort\(\)](#), et [sort\(\)](#).

10.2.24 compact

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [compact](#) (*string varname*) array *varnames*, ... |
[PHP 4]

[compact\(\)](#) accepte différents paramètres. Les paramètres peuvent être des variables contenant des chaînes, ou un tableau de chaîne, qui peut contenir d'autres tableau de noms, que [compact\(\)](#) traitera récursivement. Pour chacun des arguments, [compact\(\)](#) recherche une variable avec une variable de même nom dans la table courante des symboles, et l'ajoute dans le tableau, de manière à avoir la relation nom => 'valeur de variable'. En bref, c'est le contraire de la fonction [extract\(\)](#). Retourne le tableau ainsi créé.

Exemple avec compact()

```
<?php
$ville = "San Francisco";
$etat = "CA";
$evenement = "SIGGRAPH";
$location_vars = array("ville", "etat");
$result = compact("evenement", $location_vars);
?>
```

Après cette opération, \$result sera le tableau suivant : array(("evenement" => "SIGGRAPH", "ville" => "San Francisco", "etat" => "CA").

Voir aussi [extract\(\)](#). Note : [compact\(\)](#) a été ajoutée dans PHP 4.0.

10.2.25 count

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [count](#) (mixed *var*)
[PHP 3, PHP 4]

[count\(\)](#) retourne le nombre d'élément dans *var*, qui est généralement un tableau (et tout le reste n'aura qu'un

élément).

Retourne 1 si la variable n'est pas un tableau.

Retourne 0 si la variable n'est pas créée.

[count\(\)](#) peut retourner 0 pour une variable qui n'a pas été affectée, ou pour un tableau vide. Utilisez plutôt [isset\(\)](#) pour tester si la variable existe.

Exemple avec count()

```
<?php
$a[0] = 1;
$a[1] = 3;
$a[2] = 5;
$result = count ($a);
// $result == 3
?>
```

Voir aussi: [sizeof\(\)](#), [isset\(\)](#), et [is_array\(\)](#).

10.2.26 current

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [current](#) (array *array*)

[PHP 3, PHP 4]

Chaque tableau entretient un pointeur interne, qui est initialisé lors lorsque le premier élément est inséré dans le tableau.

[current\(\)](#) ne fait que retourner l'élément courant pointé par le pointeur interne. [current\(\)](#) ne déplace pas le pointeur. Si le pointeur est au delà du dernier élément de la liste, [current\(\)](#) retourne FALSE. Si le tableau des éléments vides ou des zéros (0 ou "", la chaîne vide) alors [current\(\)](#) retournera FALSE pour ces éléments. Il est donc impossible de déterminer si vous êtes réellement à la fin de la liste en utilisant la fonction [current\(\)](#). Pour passer en revue proprement un tableau qui peut contenir des éléments vides ou des zéros, utilisez la fonction [each\(\)](#).

Voir aussi: [end\(\)](#), [next\(\)](#), [prev\(\)](#) et [reset\(\)](#).

10.2.27 each

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [each](#) (array *array*)

[PHP 3, PHP 4]

[each\(\)](#) retourne la paire (clé/valeur) courante du tableau *array* et avance le pointeur de tableau. Cette paire est retournée dans un tableau de 4 éléments, avec les clés 0, 1, *key*, et *value*. Les éléments 0 et *key* contiennent le nom de la clé et, et 1 et *value* contiennent la valeur.

Si le pointeur interne de fichier est au delà de la fin du tableau, [each\(\)](#) retourne FALSE.

Exemples avec each()

```
<?php
```

```
$foo = array ("bob", "fred", "jussi", "jouni", "egon", "marliese");
$bar = each ($foo);
?>
```

\$bar contient maintenant les paires suivantes:

- 0 => 0
- 1 => 'bob'
- key => 0
- value => 'bob'

```
<?php
$foo = array ("Robert" => "Bob", "Seppo" => "Sepi");
$bar = each ($foo);
?>
```

\$bar contient maintenant les paires suivantes:

- 0 => 'Robert'
- 1 => 'Bob'
- key => 'Robert'
- value => 'Bob'

[each\(\)](#) est utilisé conjointement avec [list\(\)](#) pour étudier tous les éléments d'un tableau; par exemple, \$HTTP_POST_VARS:

Affichage de \$http_post_vars avec each()

```
<?php
echo "Valeurs transmises par la méthode POST:<br>";
reset ($HTTP_POST_VARS);
while (list ($key, $val) = each ($HTTP_POST_VARS)) {
echo "$key => $val<br>";
}
?>
```

Après chaque [each\(\)](#), le pointeur de tableau est déplacé au dernier éléments, ou sur le dernier élément, lorsqu'on arrive à la fin.

Voir aussi [key\(\)](#), [list\(\)](#), [current\(\)](#), [reset\(\)](#), [next\(\)](#), et [prev\(\)](#).

10.2.28 end

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[end](#) (array *array*)

[PHP 3, PHP 4]

[end\(\)](#) déplace le pointeur interne du tableau *array* jusqu'au dernier élément.

Voir aussi: [current\(\)](#), [each\(\)](#), [end\(\)](#), [next\(\)](#), et [reset\(\)](#).

10.2.29 [extract](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [extract](#) (array *var_array*, int *extract_type* , string *prefix*)

[PHP 3>= 3.0.7, PHP 4]

[extract\(\)](#) sert à exporter un tableau vers la table des symboles. Elle prend un tableau associatif *var_array* et crée les variables dont les noms sont les index de ce tableau, et leur affecte la valeur associée. Pour chaque paire clé/valeur, [extract\(\)](#) crée une variable, avec les paramètres *extract_type* et *prefix*.

Note : Depuis la version 4.0.5, [extract\(\)](#) retourne le nombre de variables extraites.

[extract\(\)](#) vérifie l'existence de la variable avant de la créer. Le traitement des collisions est déterminée par *extract_type*. Ce paramètre peut prendre une des valeurs suivantes :

EXTR_OVERWRITE

- Lors d'une collision, réécrire la variable existante.
EXTR_SKIP
- Lors d'une collision, ne pas réécrire la variable existante.
EXTR_PREFIX_SAME
- Lors d'une collision, ajouter le préfixe *prefix*, et créer une nouvelle variable.
EXTR_PREFIX_ALL
- Ajouter le préfixe *prefix*, et créer une nouvelle variable.
EXTR_PREFIX_INVALID
- Préfixer uniquement les variables aux noms invalides ou numériques avec le préfixe *prefix*. Ceci a été ajouté en PHP 4.0.5.

Si *extract_type* est omis, [extract\(\)](#) utilise EXTR_OVERWRITE par défaut.

Notez que *prefix* n'est nécessaire que pour les valeurs de *extract_type* suivantes : EXTR_PREFIX_SAME, EXTR_PREFIX_ALL ou EXTR_PREFIX_INVALID. Le résultat préfixé n'est pas un nom de variable valide, il ne sera pas importé dans la table des symboles.

[extract\(\)](#) retourne le nombre de variables réellement importées dans la table des symboles.

Une utilisation possible de [extract\(\)](#) est l'exportation vers la table des symboles de tableau de variables retourné par [wddx_deserialize\(\)](#).

Exemple avec extract()

```
<?php
/* Supposons que $var_array est un tableau retourné par
   wddx\_deserialize\(\) */
$taille = "grand";
$var_array = array("couleur" => "bleu",
                  "taille"  => "moyen",
                  "forme"  => "sphere");
extract($var_array, EXTR_PREFIX_SAME, "wddx");
print "$couleur, $taille, $forme, $wddx_taille\n";
?>
```

L'exemple ci dessus va afficher bleu, large, sphere, moyen

La variable \$taille n'a pas été réécrite, car on avait spécifié le paramètre EXTR_PREFIX_SAME, qui a permis la création \$wddx_size. Si EXTR_SKIP avait été utilisé, alors \$wddx_size n'aurait pas été créé. Avec

EXTR_OVERWRITE, \$taille aurait pris la valeur "moyen", et avec EXTR_PREFIX_ALL, les variables créées seraient \$wddx_couleur, \$wddx_taille, et \$wddx_forme.

10.2.30 in_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool in_array (mixed *needle*, array *haystack*, bool *strict*)
[PHP 4]

in_array() recherche *needle* dans *haystack* et retourne TRUE si il s'y trouve, ou FALSE sinon. Si le troisième paramètre *strict* est optionel. S'il vaut TRUE alors in_array() vérifiera aussi les types du paramètre *needle* dans *haystack*.

Exemple avec in_array()

```
<?php
$os = array("Mac", "NT", "Irix", "Linux");
if (in_array("Irix", $os))
print "Irix trouve";
?>
```

in_array() avec le paramètre strict

```
<?php
$a = array('1.10', 12.4, 1.13);
if (in_array('12.4', $a, TRUE))
echo "'12.4' trouvé avec une recherche stricte\n";
if (in_array(1.13, $a, TRUE))
echo "1.13 trouvé avec une recherche stricte\n";
?>
```

L'affichage sera :

```
1.13 trouvé avec une recherche stricte
```

Note : in_array() a été ajoutée dans PHP 4.0.

10.2.31 key

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed key (array *array*)
[PHP 3, PHP 4]

key() retourne l'index de la clé courante dans un tableau.
Voir aussi: current(), et next()

10.2.32 krsort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [krsort](#)(array *array*)

[PHP 3 >= 3.0.13, PHP 4 >= 4.0b4]

[krsort\(\)](#) trie un tableau en ordre inverse et suivant les clés, en maintenant la correspondance entre les clés et les valeurs. Cette fonction est pratique pour les tableaux associatifs.

Exemple avec krsort()

```
<?php
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
krsort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
}
?>
```

Cet exemple va afficher : fruits[d] = citron fruits[c] = ananas fruits[b] = banane fruits[a] = orange

Voir aussi [asort\(\)](#), [arsort\(\)](#), [ksort\(\)](#) [sort\(\)](#), et [rsort\(\)](#).

10.2.33 ksort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ksort](#)(array *array*)

[PHP 3, PHP 4]

[ksort\(\)](#) trie un tableau suivant les clés, en maintenant la correspondance entre les clés et les valeurs. Cette fonction est pratique pour les tableaux associatifs.

Exemple avec ksort()

```
<?php
$fruits = array("d"=>"papaye", "a"=>"orange", "b"=>"banane", "c"=>"ananas");
ksort($fruits);
reset($fruits);
while (list ($key, $val) = each ($fruits)) {
echo "$key => $val\n";
}
?>
```

Cet exemple va afficher : fruits[a] = orange fruits[b] = banane fruits[c] = ananas fruits[d] = citron

Vous pouvez modifier le comportement du tri avec les options *sort_flags*. Pour plus de détails, voyez [sort\(\)](#).

Voir aussi [asort\(\)](#), [arsort\(\)](#), [sort\(\)](#), et [rsort\(\)](#).

Note : Le second paramètre a été ajouté dans PHP 4.

10.2.34 list

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [list](#)

Tout comme [array\(\)](#), [list\(\)](#) n'est pas une véritable fonction, mais une construction syntaxique, qui permet d'assigner une série de variable en une seule ligne.

Exemple avec list()

```
<?php
<table>
  <tr>
    <th>Employee name</th>
    <th>Salary</th>
  </tr>
<?php
$result = mysql ($conn, "SELECT id, name, salary FROM employees");
while (list ($id, $name, $salary) = mysql_fetch_row ($result)) {
print ( " <tr>\n".
        " <td><a href=\"info.php3?id=$id\">$name</a></td>\n".
        " <td>$salary</td>\n".
        " </tr>\n");
}
?>
</table>
?>
```

Voir aussi: [each\(\)](#), [array\(\)](#).

10.2.35 natsort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [natsort](#) (array *array*)

[PHP 4 >= 4.0RC2]

[natsort\(\)](#) implémente un algorithme de tri qui traite les chaînes alphanumériques comme un être humain : c'est ce qui est appelé l'"ordre naturel". Un exemple de la différence de traitement entre un tel algorithme et un algorithme de tri de chaîne (comme lorsqu'on utilise [sort\(\)](#)) est illustré ci dessous :

Exemple avec natsort()

```
<?php
$array1 = $array2 = array ("img12.png", "img10.png", "img2.png", "img1.png");
sort($array1);
echo "Tri Standard\n";
print_r($array1);
natsort($array2);
echo "\nTri par Ordre Naturel\n";
print_r($array2);
?>
```

L'exemple ci dessous génère l'affichage suivant :

```
Tri Standard
Array
(
    [0] => img1.png
    [1] => img10.png
    [2] => img12.png
```

```

    [3] => img2.png
)
Tri par Ordre Naturel
Array
(
    [3] => img1.png
    [2] => img2.png
    [1] => img10.png
    [0] => img12.png
)
?>

```

Pour plus de détails, rendez vous sur le site de Martin Pool [Natural Order String Comparison](#).
Voir aussi [natcasesort\(\)](#), [strnatcmp\(\)](#) et [strnatcasecmp\(\)](#).

10.2.36 natcasesort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [natcasesort](#)(array *array*)
[PHP 4 >= 4.0RC2]

[natcasesort\(\)](#) implémente un algorithme de tri qui traite les chaînes alphanumériques comme un être humain : c'est ce qui est appelé l'"ordre naturel".

[natcasesort\(\)](#) est la version insensible à la casse de [natsort\(\)](#). Voir aussi [natsort\(\)](#) pour un exemple illustré.
Pour plus de détails, rendez vous sur le site de : Martin Pool's [Natural Order String Comparison](#).
Voir aussi [sort\(\)](#), [natsort\(\)](#), [strnatcmp\(\)](#) et [strnatcasecmp\(\)](#).

10.2.37 next

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [next](#)(array *array*)
[PHP 3, PHP 4]

[next\(\)](#) retourne l'élément suivant du tableau, ou FALSE si il n'y a plus d'éléments. Le pointeur de interne de tableau est avancé d'un élément.

[next\(\)](#) se comporte comme [current\(\)](#), mais avec une différence : il avance le pointeur interne de tableau d'un élément avant de retourner la valeur qu'il pointe. Lorsque le pointeur dépasse le dernier élément, [next\(\)](#) retourne FALSE. Si le tableau contient des éléments vides ou des zéros, [next\(\)](#) retournera FALSE pour ces éléments. Pour passer proprement en revue un tableau, il faut utiliser [each\(\)](#).

Voir aussi: [current\(\)](#), [end\(\)](#), [prev\(\)](#) et [reset\(\)](#).

10.2.38 pos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [pos](#)(array *array*)
[PHP 3, PHP 4]

[pos\(\)](#) est une fonction alias de [current\(\)](#).
Voir aussi: [end\(\)](#), [next\(\)](#), [prev\(\)](#) et [reset\(\)](#).

10.2.39 prev

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [prev](#)(array *array*)

[PHP 3, PHP 4]

[prev\(\)](#) repositionne le pointeur interne de tableau à la dernière place qu'il occupait, ou bien retourne FALSE si il ne reste plus d'éléments. Si le tableau contient des éléments vides, [prev\(\)](#) retournera FALSE pour ces éléments aussi. Pour passer en revue tous les éléments, utilisez plutôt [each\(\)](#).

[prev\(\)](#) se comporte exactement comme [next\(\)](#), mais il fait reculer le pointeur plutôt que de l'avancer.

Voir aussi: [current\(\)](#), [end\(\)](#), [next\(\)](#) et [reset\(\)](#).

10.2.40 range

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [range](#)(int *low*, int *high*)

[PHP 3 >= 3.0.8, PHP 4 >= 4.0b4]

[range\(\)](#) retourne un tableau contenant tous les entiers depuis *low* jusqu'à *high*, inclus.

Voir [shuffle\(\)](#) pour un exemple d'utilisation.

10.2.41 reset

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [reset](#)(array *array*)

[PHP 3, PHP 4]

[reset\(\)](#) replace le pointeur de tableau *array* au premier élément.

[reset\(\)](#) retourne la valeur du premier élément.

Voir aussi: [current\(\)](#), [each\(\)](#), [next\(\)](#), [prev\(\)](#), et [reset\(\)](#).

10.2.42 rsort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [rsort](#)(array *array*)

[PHP 3, PHP 4]

[rsort\(\)](#) effectue un tri en ordre décroissant (du plus grand au plus petit).

Exemple avec [rsort\(\)](#)

```
<?php
$fruits = array("papaye","orange","banane","ananas");
rsort($fruits);
for (reset($fruits); list($key,$value) = each($fruits); ) {
echo "fruits[$key] = ", $value, "\n";
}
?>
```

Cet exemple va afficher: fruits[0] = papaye fruits[1] = orange fruits[2] = banane fruits[3] = ananas Les fruits ont été classés dans l'ordre alphabétique inverse.
Voir aussi: [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [sort\(\)](#), et [usort\(\)](#).

10.2.43 shuffle

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [shuffle](#)(array *array*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[shuffle\(\)](#) mélange les éléments d'un tableau.

Exemple avec shuffle()

```
<?php
$numbers = range (1,20);
srand (time());
shuffle ($numbers);
while (list(, $number) = each ($numbers)) {
echo "$number ";
}
?>
```

Voir aussi [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), [sort\(\)](#) et [usort\(\)](#).

10.2.44 sizeof

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sizeof](#)(array *array*)

[PHP 3, PHP 4]

[sizeof\(\)](#) retourne le nombre d'élément d'un tableau.

Voir aussi: [count\(\)](#)

10.2.45 sort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sort](#)(array *array*)

[PHP 3, PHP 4]

[sort\(\)](#) trie le tableau *array*. Les éléments seront triés du plus petit au plus grand.

Exemple avec sort()

```
<?php
$fruits = array("papaye","orange","banane","ananas");
sort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
echo "fruits[$key] = ".$fruits[$key]."\n";
}
?>
```

Cet exemple va afficher : fruits[0] = ananas fruits[1] = banane fruits[2] = orange fruits[3] = papaye Les fruits ont été classé dans l'ordre alphabétique.

Voir aussi: [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), et [usort\(\)](#).

10.2.46 uasort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [uasort](#) (array *array*, function *cmp_function*)

[PHP 3>= 3.0.4, PHP 4]

[uasort\(\)](#) trie un tableau en conservant la correspondance entre les index et leurs valeurs. [uasort\(\)](#) sert essentiellement lors de tri de tableaux associatifs où l'ordre des éléments est significatif. La fonction de comparaison utilisée est définie par l'utilisateur.

10.2.47 uksort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [uksort](#) (array *array*, function *cmp_function*)

[PHP 3>= 3.0.4, PHP 4]

[uksort\(\)](#) trie les clés du tableau en utilisant une fonction définie par l'utilisateur. Si un tableau qui doit être trié avec un critère complexe, il est préférable d'utiliser [uksort\(\)](#).

Exemple avec uksort()

```
<?php
function mycompare($a, $b) {
    if ($a == $b) return 0;
    return ($a > $b) ? -1 : 1;
}
$a = array(4 => "quatre", 3 => "trois", 20 => "vingt", 10 => "dix");
uksort($a, mycompare);
while(list($key, $value) = each($a)) {
    echo "$key: $value\n";
}
?>
```

Cet exemple affichera: 20: vingt 10: dix 4: quatre 3: trois

Voir aussi: [arsort\(\)](#), [asort\(\)](#), [uasort\(\)](#), [ksort\(\)](#), [rsort\(\)](#), et [sort\(\)](#).

10.2.48 usort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [usort](#) (array *array*, function *cmp_function*)

[PHP 3>= 3.0.3, PHP 4]

[usort\(\)](#) va trier un tableau avec ses valeurs, en utilisant une fonction définie par l'utilisateur. Si un tableau doit être trié avec un critère complexe, il est préférable d'utiliser cette méthode.

La fonction de comparaison *cmp_function* doit retourner un entier, qui sera inférieur, égal ou supérieur à zéro suivant que le premier argument est considéré comme plus petit, égal ou plus grand que le second argument. Si les deux arguments sont égaux, leur ordre est indéfini.

Exemple avec usort()

```
<?php
function cmp($a,$b) {
if ($a == $b) return 0;
return ($a > $b) ? -1 : 1;
}
$a = array(3,2,5,6,1);
usort($a, "cmp");
while(list($key,$value) = each($a)) {
echo "$key: $value\n";
}
?>
```

Cet exemple va afficher : 0 : 6 1 : 5 2 : 3 3 : 2 4 : 1 Note : *Evidemment dans ce cas trivial, [rsort\(\)](#) serait plus approprié.*

Les bibliothèques de tri rapides sur lesquelles reposent PHP peuvent le conduire à un plantage, si la fonction de comparaison ne retourne pas une valeur cohérente.

Voir aussi: [arsort\(\)](#), [asort\(\)](#), [ksort\(\)](#), [rsort\(\)](#) et [sort\(\)](#).

10.3 Aspell

[\[Notes en ligne\]](#)

Les fonctions Aspell vous permettent de vérifier l'orthographe d'un mot, et d'offrir des suggestions de corrections. Plusieurs langues sont disponibles, comme le franç, l'allemand, le suédois et le danois.

Note : *aspell fonctionne avec de très vieilles versions (jusqu'à la version .27.* ou presque) de la librairie aspell. Ce module, et ces versions d'Aspell ne sont plus supportées. Si vous voulez utiliser les possibilités de vérifications d'orthographe en PHP, utilisez plutôt [10.58 Pspell](#). Ce module utilise la librairie pspell qui fonctionne avec les nouvelles versions de Aspell.*

Vous avez besoin de la librairie Aspell, disponible à : <http://aspell.sourceforge.net/>.

10.3.1 aspell_new

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [aspell_new](#) (string *master*, string *personal*)

[PHP 3>= 3.0.7, PHP 4]

[aspell_new\(\)](#) ouvre un nouveau dictionnaire, et retourne un identifiant de dictionnaire pour utilisation ultérieure dans les fonctions aspell.

aspell_new

```
<?php
$aspell_link=aspell_new( "english" );
?>
```


10.3.2 aspell_check

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [aspell_check](#)(int *dictionary_link*, string *word*)

[PHP 3>= 3.0.7, PHP 4]

[aspell_check\(\)](#) vérifie l'orthographe d'un mot et retourne TRUE si l'orthographe est correcte, et FALSE sinon.

aspell_check

```

<?php
$aspell_link=aspell_new("english");
if (aspell_check($aspell_link,"testt")) {
echo "L'orthographe est correcte.";
} else {
echo "Désolé, l'orthographe est incorrecte.";
}
?>

```

10.3.3 aspell_check_raw

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [aspell_check_raw](#)(int *dictionary_link*, string *word*)

[PHP 3>= 3.0.7, PHP 4]

[aspell_check_raw\(\)](#) vérifie l'orthographe d'un mot sans en changer la casse, et sans essayer de supprimer les espaces aux extrémités. Elle retourne TRUE si l'orthographe est bonne, et FALSE sinon.

aspell_check_raw

```

<?php
$aspell_link=aspell_new("french");
if (aspell_check_raw($aspell_link,"testt")) {
echo "L'orthographe est OK";
} else {
echo "Attention : faute d'orthographe";
}
?>

```

10.3.4 aspell_suggest

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [aspell_suggest](#)(int *dictionary_link*, string *word*)

[PHP 3>= 3.0.7, PHP 4]

[aspell_suggest\(\)](#) retourne un tableau contenant les orthographes possibles d'un mot mal formé.

aspell_suggest

```
<?php
$aspell_link=aspell_new("french");
if (!aspell_check($aspell_link,"testt")) {
    $suggestions=aspell_suggest($aspell_link,"testt");
    for($i=0; $i < count($suggestions); $i++) {
        echo "Orthographes envisageables : " . $suggestions[$i] . "<br>";
    }
}
?>
```

10.4 Nombres de grande taille

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que si l'option de configuration `--enable-bcmath` a été activée lors de la compilation.

Note : Suite aux changement de licence, la librairie *BCMATH* est désormais distribuée séparément. Vous pouvez télécharger l'archive à <http://www.php.net/extra/number4.tar.gz>. Lisez attentivement le fichier ``README.BCMATH'` de la distribution PHP.

10.4.1 bcadd

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string **bcadd** (string *left operand*, string *right operand*, int *scale*)

[PHP 3, PHP 4]

bcadd() additionne *left operand* avec l'opérande *right operand* et renvoie la somme sous forme de chaîne de caractères. Le paramètre optionnel *scale* est utilisé pour définir le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcsub\(\)](#).

10.4.2 bccomp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int **bccomp** (string *left operand*, string *right operand*, int *scale*)

[PHP 3, PHP 4]

bccomp() compare l'opérande *left operand* avec l'opérande *right operand* et renvoie le résultat sous forme de valeur numérique (integer). Le paramètre optionnel *scale* est utilisé pour définir le nombre de chiffres après la virgule utilisés lors de la comparaison. Le résultat est 0 si les deux opérandes sont égales. Si l'opérande *left operand* est plus grande que l'opérande *right operand*, le résultat est 1. Si l'opérande *left operand* est plus petite que l'opérande *right operand*, le résultat est -1.

10.4.3 bcdiv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bctdiv](#) (string *left operand*, string *right operand*, int *scale*)
[PHP 3, PHP 4]

[bctdiv\(\)](#) divise l'opérande *left operand* par l'opérande *right operand* et renvoie le résultat. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.
Voir aussi [bcmul\(\)](#).

[10.4.4 bctmod](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bctmod](#) (string *left operand*, string *modulus*)
[PHP 3, PHP 4]

[bctmod\(\)](#) retourne le reste de la division entre *left operand* en utilisant *modulus*.
Voir aussi [bctdiv\(\)](#).

[10.4.5 bcmul](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcmul](#) (string *left operand*, string *right operand*, int *scale*)
[PHP 3, PHP 4]

[bcmul\(\)](#) multiplie l'opérande *left operand* par l'opérande *right operand* et renvoie le résultat. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.
Voir aussi [bctdiv\(\)](#).

[10.4.6 bcpow](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcpow](#) (string *x*, string *y*, int *scale*)
[PHP 3, PHP 4]

[bcpow\(\)](#) élève *x* à la puissance *y*. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.
Voir aussi [bcsqrt\(\)](#).

[10.4.7 bcscale](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcscale](#) (int *scale*)
[PHP 3, PHP 4]

[bcscale\(\)](#) définit la précision par défaut pour toutes les fonctions mathématiques sur des nombres de taille arbitraire qui suivent et qui omettent le paramètre *scale*.

10.4.8 bcsqrt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcsqrt](#) (string *operand*, int *scale*)
[PHP 3, PHP 4]

[bcsqrt\(\)](#) renvoie la racine carrée de l'opérande *operand*. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcpow\(\)](#).

10.4.9 bcsub

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bcsub](#) (string *left operand*, string *right operand*, int *scale*)
[PHP 3, PHP 4]

[bcsub\(\)](#) soustrait l'opérande *right operand* de l'opérande *left operand* et renvoie le résultat sous forme de chaîne de caractères. Le paramètre optionnel *scale* définit le nombre de chiffres après la virgule dans le résultat.

Voir aussi [bcadd\(\)](#).

10.5 Calendrier

[\[Notes en ligne\]](#)

Les fonctions de calendrier ne sont disponibles que si l'extension calendrier a été compilée. Elle est située dans les sous-dossiers "dl" ou "ext" de votre distribution de PHP. Lisez le fichier README pour plus de détails.

L'extension de calendrier propose une série de fonctions qui simplifie les conversions entre les différents formats de calendrier. La référence est le nombre de jour du calendrier Julien. C'est le nombre de jours depuis une date qui commence bien au delà des dates les plus reculées dont on a besoin (située en 4000 avant J.C.). Pour convertir une date d'un calendrier à un autre, il faut d'abord la convertir dans ce calendrier, puis convertir le résultat dans le calendrier désiré. Attention, le nombre de jour du calendrier Julien est un système très différent du calendrier Julien!. Pour plus d'informations (en anglais), reportez vous à

<http://genealogy.org/~scottlee/cal-overview.html>. Les traductions issues de ces pages seront mises entre guillemets.

10.5.1 jdtogregorian

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtogregorian](#) (int *julianday*)
[PHP 3, PHP 4]

[jdtogregorian\(\)](#) convertit le nombre de jours du calendrier Julien en une chaîne contenant une date du calendrier grégorien, au format "mois/jour/année".

[10.5.2 gregoriantojd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gregoriantojd](#) (int *month*, int *day*, int *year*)

[PHP 3, PHP 4]

Intervalle de validité pour le calendrier grégorien : 4714 avant JC à 9999 après JC.A.D.

Bien qu'il soit possible de manipuler des dates jusqu'en 4714 avant JC, une telle utilisation n'est pas significative. En effet, ce calendrier fut créé le 18 octobre 1582 après J.C. (ou 5 octobre 1582 en calendrier grec). Certains pays ne l'acceptèrent que bien plus tard. Par exemple, les britanniques n'y passèrent en 1752, les Russes en 1918 et les Grecs en 1923. La plus part des pays européens utilisaient le calendrier Julien avant le Grégorien.

Fonctions calendrier

```
<?php
$jd = gregoriantojd(10,11,1970);
echo("$jd\n");
$gregorian = jdtogregorian($jd);
echo("$gregorian\n");
?>
```

[10.5.3 jdtojulian](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtojulian](#) (int *julianday*)

[PHP 3, PHP 4]

[jdtojulian\(\)](#) convertit le nombre de jours du calendrier Julien en une chaîne contenant la date du calendrier Julien, au format "mois/jour/année".

[10.5.4 juliantojd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [juliantojd](#) (int *month*, int *day*, int *year*)

[PHP 3, PHP 4]

Intervalle de validité du calendrier Julien : 4713 avant JC à 9999 après J.C..

Bien qu'il soit possible de manipuler des dates jusqu'en 4713 avant JC, une telle utilisation n'est pas significative. En effet, ce calendrier fut créé en 46 avant JC, et ses détails ne furent finalisés qu'au plus tôt en 8 après JC, et probablement pas avant le 4ème siècle après JC. De plus, le début de l'année variait suivant les peuples, certains n'acceptant pas janvier comme premier mois de l'année.

[10.5.5 jdtojewish](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtojewish](#) (int *julianday*)

[PHP 3, PHP 4]

[jdtowishO](#) convertit le nombre de jours du calendrier julien en date du calendrier juif.

10.5.6 [jewishtojd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [jewishtojd](#) (int *month*, int *day*, int *year*)

[PHP 3, PHP 4]

Bien qu'il soit possible de manipuler des dates à partir de l'an 1 (3761 avant JC), une telle utilisation a peu de sens.

Le calendrier juif a été utilisé depuis plusieurs dizaines de siècles, mais dans les premiers temps, il n'y avait pas de formule pour déterminer le début du mois. Un nouveau mois commençait quand une nouvelle lune était observée.

10.5.7 [jdtofrrench](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdtofrrench](#) (int *juliandaycount*)

[PHP 3, PHP 4]

[jdtofrrench\(\)](#) convertit le nombre de jours du calendrier julien en date du calendrier français républicain.

10.5.8 [frenchtojd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [frenchtojd](#) (int *month*, int *day*, int *year*)

[PHP 3, PHP 4]

[frenchtojd\(\)](#) convertit une date du calendrier français républicain en nombre de jour du calendrier julien.

Ces fonctions convertissent les dates comprises entre l'an 1 et l'an 14 (22 September 1792 à 22 September 1806 en calendrier grégorien). Cela couvre plus que la durée d'existence de ce calendrier.

10.5.9 [jdmonthname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [jdmonthname](#) (int *julianday*, int *mode*)

[PHP 3, PHP 4]

[jdmonthname\(\)](#) retourne une chaîne contenant le nom du mois. *mode* indique de quel calendrier dépend ce mois, et quel type de nom doit être retourné.

Mode	Signification
0	Grégorien – abrégé
1	Grégorien
2	Julien – abrégé

3	Julien
4	Juif
5	Républicain français

10.5.10 [jddayofweek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [jddayofweek](#) (int *julianday*, int *mode*)

[PHP 3, PHP 4]

[jddayofweek\(\)](#) retourne le numéro du jour de la semaine. Peut retourner une chaîne ou un entier, en fonction du mode.

Mode	Signification
0	Retourne le numéro du jour comme un entier (0=dimanche, 1=lundi, etc.) @tab
1	Retourne une chaîne contenant le nom du jour (anglais grégorien) @tab
2	Retourne une chaîne contenant le nom abrégé du jour de la semaine (anglais grégorien). @tab

10.5.11 [easter_date](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [easter_date](#) (int *year*)

[PHP 3 >= 3.0.9, PHP 4 >= 4.0RC2]

[easter_date\(\)](#) retourne un timestamp UNIX pour Pâques, à minuit, pour une année donnée. Si l'année n'est pas précisée, c'est l'année en cours qui est utilisée.

ATTENTION: [easter_date\(\)](#) génère une alerte (Warning) si la date tombe hors de la zone de validité des timestamp UNIX (i.e. avant 1970 ou après 2037).

Exemples avec [easter_date\(\)](#)

```
echo date( "M-d-Y", easter_date(1999) );           /* "04 avril 1999" */
echo date( "M-d-Y", easter_date(2000) );           /* "23 avril 2000" */
echo date( "M-d-Y", easter_date(2001) );           /* "15 avril 2001" */
```

La date de Pâques a été fixée par le concile de Nicée, en 325 de notre ère, comme étant le dimanche après la première lune pleine qui suit l'équinoxe de printemps. L'équinoxe de printemps est considéré comme étant toujours le 21 mars, ce qui réduit le problème au calcul de la date de la lune pleine qui suit, et le dimanche suivant. L'algorithme fut introduit vers 532, par Dionysius Exiguus. Avec le calendrier Julien, (pour les années avant 1753), un cycle de 19 ans suffit pour connaître les date des phases de la lune. Avec le calendrier grégorien, (à partir des années 1753, conçu par Clavius et Lilius, puis introduit par le pape Gregoire XIII en Octobre 1582, et en Grande Bretagne et ses colonies en septembre 1752), deux facteurs de corrections ont été ajoutés pour rendre le cycle plus précis.

(Ce code est basé sur le programme en C de Simon Kershaw, <webmaster@ely.anglican.org>)
 Voir [easter_days\(\)](#) pour les calculs de date de Pâques avant 1970 et après 2037.

10.5.12 [easter_days](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [easter_days](#)(int *year*)

[PHP 3 >= 3.0.9, PHP 4 >= 4.0RC2]

[easter_days\(\)](#) retourne le nombre de jour entre le 21 Mars et Pâques, pour une année donnée. Si l'année n'est pas précisée, l'année en cours est utilisée par défaut.

[easter_days\(\)](#) peut être utilisée à la place de [easter_date\(\)](#) pour calculer la date de Pâques, pour les années qui tombent hors de l'intervalle de validité des timestamps UNIX (i.e. avant 1970 ou après 2037).

Exemple [easter_date\(\)](#)

```
<?php
echo easter_days(1999);          /* 14, i.e. 4 Avril   */
echo easter_days(1492);          /* 32, i.e. 22 Avril  */
echo easter_days(1913);          /* 2, i.e. 23 Mars    */
?>
```

La date de Pâques a été fixée par le concile de Nicée, en 325 de notre ère, comme étant le dimanche après la première lune pleine qui suit l'équinoxe de printemps. L'équinoxe de printemps est considéré comme étant toujours le 21 mars, ce qui réduit le problème au calcul de la date de la lune pleine qui suit, et le dimanche suivant. L'algorithme fut introduit vers 532, par Dionysius Exiguus. Avec le calendrier Julien, (pour les années avant 1753), un cycle de 19 ans suffit pour connaître les date des phases de la lune. Avec le calendrier grégorien, (à partir des années 1753, conçu par Clavius et Lilius, puis introduit par le pape Gregoire XIII en Octobre 1582, et en Grande Bretagne et ses colonies en septembre 1752), deux facteurs de corrections ont été ajoutés pour rendre le cycle plus précis.

(Ce code est basé sur le programme en C de Simon Kershaw, <webmaster@ely.anglican.org>)

Voir aussi [easter_date\(\)](#).

10.5.13 [unixtojd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [unixtojd](#)(int *timestamp*)

[PHP 4 >= 4.0RC2]

[unixtojd\(\)](#) retourne le nombre de jours juliens du timestamp UNIX *timestamp* (nombre de secondes depuis le 1/1/1970), ou pour le jour courant si *timestamp* est omis.

Voir aussi [jdtounix\(\)](#).

Note : [unixtojd\(\)](#) n'est disponible qu'à partir de la version PHP 4.0RC1.

10.5.14 [jdtounix](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [jdtounix](#)(int *jday*)

[PHP 4 >= 4.0RC2]

[`jdtounix\(\)`](#) retourne un timestamp UNIX correspondant au nombre de jour julien *jday* ou FALSE si *jday* n'est pas dans l'intervalle de validité de l'époque UNIX. (années grégorienne entre 1970 et 2037 ou 2440588 <= *jday* <= 2465342).

Voir aussi [`jdtounix\(\)`](#).

Note : [`jdtounix\(\)`](#) n'est disponible qu'à partir de la version PHP 4.0RC1.

10.6 Paiement CCVS

[\[Notes en ligne\]](#)

Ces fonctions font l'interface avec les API CCVS, vous permettant de travailler directement avec CCVS depuis vos scripts PHP. CCVS est la solution apportée par [RedHat](#) au problème de l'intermédiaire, lors du traitement de transactions de cartes de crédit. Il vous permet travailler directement avec les maisons de crédits, via votre boîte *nix et un modem. En utilisant le module CCVS pour PHP, vous pouvez effectuer des transactions avec les cartes de crédits, directement depuis vos scripts PHP via CCVS. La suite vous montrera comment procéder.

Pour activer le support CCVS de PHP, commencez par vérifier votre installation CCVS. Vous devez configurer PHP avec l'option `--with-ccvs`. Si vous utilisez cette option sans spécifier le chemin de votre installation, PHP essaiera de la trouver à sa position par défaut (`/usr/local/ccvs`). Si CCVS est installé dans un autre dossier, lancez la configuration avec : `--with-ccvs=$ccvs_path`, où `$ccvs_path` est le chemin de votre installation CCVS. Notez bien que CCVS requiert que `$ccvs_path/lib` et `$ccvs_path/include` existent, et qu'ils contiennent respectivement `cv_api.h` et `libccvs.a` sous `include` et `lib`.

De plus, un démon `ccvsd` doit être disponible sur votre configuration, et qu'il soit accessible à vos scripts PHP. Assurez vous aussi que l'utilisateur qui exécute les scripts PHP est le même que celui qui a installé CCVS (i.e. si vous avez installé CCVS avec l'utilisateur 'ccvs', vos scripts PHP doivent tourner aussi en 'ccvs').

Plus de détails sur CCVS sont disponibles à <http://www.redhat.com/products/ccvs>.

Cette documentatin est en chantier. Jusqu'à sa finalisation, RedHat entretient une version légèrement démodée mais bien pratique à <http://www.redhat.com/products/ccvs/support/CCVS3.3docs/ProgPHP.html>.

@xref{function. , , () }

10.7 Objets

[\[Notes en ligne\]](#)

10.7.1 Introduction

[\[Notes en ligne\]](#)

10.7.1.1 About

[\[Notes en ligne\]](#)

Ces fonctions permettent de gérer les classes et les objets. Vous pouvez notamment connaître le nom de la classe d'un objet, ses membres et ses méthodes, et tous les objets parents (les classes qui sont étendues par la classe d'un objet).

10.7.1.2 Exemple d'utilisation

[\[Notes en ligne\]](#)

Dans cet exemple, on définit une classe de base, et une extension. La classe de base définit un légume, si il est mangeable ou pas, et sa couleur. La sous-classe *epinard* ajoute une méthode pour le cuisiner, et une autre

pour savoir s'il est cuisiné.

classes.inc

```
<?php
// classe de base, avec ses membres et ses méthodes
class Legume {
var $mangeable;
var $couleur;
function legume( $mangeable, $couleur="green" ) {
    $this->mangeable = $mangeable;
    $this->couleur = $couleur;
}
function est_mangeable() {
return $this->mangeable;
}
function quelle_couleur() {
return $this->couleur;
}
} // fin de la classe Legume
// extend la classe de base
class Epinard extends Legume {
var $cuit = FALSE;
function Epinard() {
    $this->Legume( TRUE, "green" );
}
function cuisine() {
    $this->cuit = TRUE;
}
function est_cuit() {
return $this->cuit;
}
} // fin de la classe Epinard
?>
```

Lorsqu'on instancie deux objets de ces classes, et qu'on affiche leurs informations, y compris leur héritage. On définit ici des utilitaires qui servent essentiellement à afficher ces informations proprement.

test_script.php

```
<pre>
<?php
include "classes.inc";
// utilitaires
function print_vars($obj) {
    $arr = get_object_vars($obj);
    while (list($prop, $val) = each($arr))
        echo "\t$prop = $val\n";
}
function print_methods($obj) {
    $arr = get_class_methods(get_class($obj));
    foreach ($arr as $method)
        echo "\tfunction $method()\n";
}
function class_parentage($obj, $class) {
    global $$obj;
    if (is_subclass_of($$obj, $class)) {
        echo "L'objet $obj belongs to class ".$get_class($$obj);
        echo " est une sous-classe de $class\n";
    }
}
```

```

    } else {
echo "L'objet $obj n'est pas une sous classe $class\n";
    }
}
// instantie 2 objets
$legume = new Legume(TRUE,"blue");
$feuilles = new Epinard();
// affiche les informations sur ces objets
echo "legume: CLASS ".get_class($legume)."\n";
echo "feuilles: CLASS ".get_class($feuilles);
echo ", PARENT ".get_parent_class($feuilles)."\n";
// affiche les propriétés du légume
echo "\nLégume: Propriétés \n";
print_vars($legume);
// et les méthodes de "feuilles"
echo "\nfeuilles: Methods\n";
print_methods($feuilles);
echo "\nParentée:\n";
class_parentage("feuilles", "Epinard");
class_parentage("feuilles", "Legume");
?>
</pre>

```

Il est important de noter que les exemples ci-dessus, les objets *\$feuilles* sont une instance de Epinard est une sous classe de Legume, donc la dernière partie du script va afficher :

```
[...] Parentée: L'objet feuilles n'est pas une sous classe Spinach
L'objet feuilles est une sous-classe de Legume
```

10.7.2 [get_declared_classes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_declared_classes](#)(void)

[PHP 4 >= 4.0RC2]

[get_declared_classes\(\)](#) retourne un tableau contenant la liste des fonctions déclarées dans le script courant.

Note : En PHP 4.0.1pl2, trois classes supplémentaires sont retournées, au début de ce tableau : *stdClass* (définie dans `Zend/zend.c`), *OverloadedTestClass* (définie dans `ext/standard/basic_functions.c`) et *Directory* (définie dans `ext/standard/dir.c`).

10.7.3 [call_user_method](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [call_user_method](#)(string *method_name* , object *obj* , mixed *parameter* , mixed ...)

[PHP 3 >= 3.0.3, PHP 4]

Appelle la méthode *method_name* depuis l'objet *obj*. Un exemple d'utilisation de cet objet est présenté ci-dessous, où une classe est définie, puis instantiée. On utilise alors [call_user_method\(\)](#) pour appeler indirectement les méthodes *print_info*.

```
<?php
class Pays {
```

```

var $NOM;
var $TLD;
function Pays($nom, $tld) {
    $this->NOM = $nom;
    $this->TLD = $tld;
}
function print_info($prestr="") {
echo $prestr."Pays: ".$this->NOM."\n";
echo $prestr."Nom de domaine: ".$this->TLD."\n";
}
}
$unPays = new Pays("Pérou","pe");
echo "* Appel de la méthode directement\n";
$cntry->print_info();
echo "\n* Appel de la méthode indirectement\n";
call_user_method ("print_info", $unPays, "\t");
?>

```

See also [call_user_func\(\)](#).

10.7.4 class_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [class_exists](#) (string *class_name*)
 [PHP 4 >= 4.0b4]

[class_exists\(\)](#) retourne TRUE si la classe *class_name* a été définie, et FALSE sinon.

10.7.5 get_class

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_class](#) (object *obj*)
 [PHP 4 >= 4.0b2]

[get_class\(\)](#) retourne la classe de l'objet *obj*.
 Voir aussi [get_parent_class\(\)](#), [is_subclass_of\(\)](#).

10.7.6 get_class_methods

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_class_methods](#) (string *class_name*)
 [PHP 4 >= 4.0RC1]

[get_class_methods\(\)](#) retourne un tableau contenant les noms des méthodes de la classe *class_name*.
 Voir aussi [get_class_vars\(\)](#), [get_object_vars\(\)](#).

10.7.7 get_class_vars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_class_vars](#) (string *class_name*)

[PHP 4 >= 4.0RC1]

[get_class_vars\(\)](#) retourne un tableau contenant les valeurs par défaut des attributs de la classe *class_name*.

10.7.8 [get_object_vars](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)array [get_object_vars](#)(object *obj*)

[PHP 4 >= 4.0RC1]

[get_object_vars\(\)](#) retourne un tableau associatif contenant les propriétés de l'objet *obj*. Les clés du tableau sont les noms des propriétés de l'objet. Si des variables déclarées dans la classe de l'objet *obj*, n'ont pas été assignées, elles ne seront pas retournées dans le tableau.**Exemple avec [get_object_vars\(\)](#)**

```

<?php
class Point2D {
var $x, $y;
var $nom;
function Point2D($x, $y) {
    $this->x = $x;
    $this->y = $y;
}
function donne_nom($nom) {
    $this->nom = $nom;
}
function LitPoint() {
return array("x" -> $this->x,
            "y" -> $this->y,
            "nom" -> $this->nom);
}
}
$p1 = new Point2D(1.233, 3.445);
print_r(get_object_vars($p1));
// "$nom" est déclaré, mais non défini
// Array
// (
//     [x] -> 1.233
//     [y] -> 3.445
// )
$p1->setnom("point #1");
print_r(get_object_vars($p1));
// Array
// (
//     [x] -> 1.233
//     [y] -> 3.445
//     [nom] -> point #1
// )
?>

```

Voir aussi [get_class_methods\(\)](#), [get_class_vars\(\)](#)

10.7.9 get_parent_class

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_parent_class](#)(object *obj*)
[PHP 4 >= 4.0b2]

[get_parent_class\(\)](#) retourne le nom de la classe de l'objet *obj*.
Voir aussi [get_class\(\)](#), [is_subclass_of\(\)](#)

10.7.10 is_subclass_of

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_subclass_of](#)(object *obj*, string *superclass*)
[PHP 4 >= 4.0b4]

[is_subclass_of\(\)](#) retourne TRUE si l'objet *obj* est une sous-classe de *superclass*, FALSE sinon.
Voir aussi [get_class\(\)](#), [get_parent_class\(\)](#)

10.7.11 method_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [method_exists](#)(object *object*, string *method_name*)
[PHP 4 >= 4.0b2]

[method_exists\(\)](#) retourne TRUE si la méthode *method_name* a été définie pour la classe *object*, et sinon, retourne FALSE.

10.8 Support COM pour Windows

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que sous les versions Windows de PHP. Elles ont été ajoutées dans PHP 4.

10.8.1 com_load

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [com_load](#)(string *module name*, string *server name*)
[PHP 3>= 3.0.3, PHP 4]

[com_load\(\)](#) crée un nouveau composant COM, et retourne une référence dessus. Retourne FALSE en cas d'échec.

10.8.2 com_invoke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [com_invoke](#)(resource *com_object*, string *function_name*, mixed *function*

parameters, ...)

[PHP 3>= 3.0.3, PHP 4]

[com_invoke\(\)](#) appelle la méthode *function_name* du composant COM *com_object*. Retourne FALSE en cas d'erreur, sinon retourne le résultat de la fonction *function_name* en cas de succès.

10.8.3 com_propget

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [com_propget](#)(resource *com_object*, string *property*)

[PHP 3>= 3.0.3, PHP 4]

[com_propget\(\)](#) est un alias de [com_get\(\)](#).

10.8.4 com_get

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [com_get](#)(resource *com_object*, string *property*)

[PHP 3>= 3.0.3, PHP 4]

[com_get\(\)](#) retourne la valeur de la propriété *property* du composant COM *com_object*. Retourne FALSE en cas d'erreur.

10.8.5 com_propput

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [com_propput](#)(resource *com_object*, string *property*, mixed *value*)

[PHP 3>= 3.0.3, PHP 4]

[com_propput\(\)](#) est un alias de [com_set\(\)](#).

10.8.6 com_propset

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [com_propset](#)(resource *com_object*, string *property*, mixed *value*)

[PHP 3>= 3.0.3, PHP 4]

Cette fonction est un alias de [com_propput\(\)](#).

10.8.7 com_set

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [com_set](#)(resource *com_object*, string *property*, mixed *value*)

[PHP 3>= 3.0.3, PHP 4]

Remplace la valeur de la propriété *property* du composante COM *com_object* par *value*. Retourne TRUE en cas de succès, et FALSE sinon.

10.9 ClibPDF

[\[Notes en ligne\]](#)

ClibPDF vous permet de créer des documents PDF avec PHP. Cette librairie est disponible à [FastIO](#) mais n'est pas gratuite. Vous devez lire la licence avant de l'utiliser. Si vous ne pouvez pas accepter la licence, essayez plutôt `pdflib` de Thomas Merz, qui est aussi très puissante. Les fonctionnalités de ClibPDF et ses API sont très similaires à celles de Thomas Merz's `pdflib` mais, selon `FastIO`, ClibPDF est plus rapide, et crée des documents plus compacts. Cela peut avoir changé depuis la version 2.0 de `pdflib`. Un test de vitesse (avec `pdfclock.c` issue des exemples de `pdflib` 2.0 transformé en script PHP) ne montre aucune différence de vitesse. La taille des fichiers est similaire si la compression n'est pas utilisée. Il vaut mieux alors essayer les deux, et choisir celui qui vous convient le mieux.

Cette documentation devrait être lue avec le manuel ClibPDF sous la main, car il est beaucoup plus détaillé. Beaucoup de fonctions sont natives de ClibPDF et se retrouvent dans le module PHP, et tout comme `pdflib`, elles ont le même nom. Toutes les fonctions, hormis [cpdf_open\(\)](#) utilisent un pointeur sur un document comme premier paramètre. Actuellement, ce pointeur n'est pas utilisé en interne, car ClibPDF ne supporte pas la création de plusieurs documents PDF simultanément. En fait, il ne vaut mieux pas l'envisager, car les résultats sont aléatoires. Je ne veux même pas imaginer les problèmes qui pourrait se poser avec les environnements multi-tâches. Selon l'auteur de ClibPDF, cette situation va changer dans les prochaines versions (lorsque cette documentation a été traduite, c'était la version 1.10). Si vous avez besoin de cette fonctionnalité, utilisez `pdflib`.

Note : La fonction [cpdf_set_font\(\)](#) a changé depuis le PHP 3.0 pour supporter les polices asiatiques. Le paramètre d'encodage n'est plus un entier, mais une chaîne.

Un des gros avantages de ClibPDF sur `pdflib` est la possibilité de créer complètement un document sans passer par des fichiers temporaires. Il est aussi possible d'utiliser des coordonnées avec une unité de longueur prédéfinie. C'est une fonctionnalité bien pratique mais qui peut être simulée avec [pdf_translate\(\)](#).

Un autre atout de ClibPDF est que chaque page peut être modifiée à tout moment même si une nouvelle page a été ouverte. La fonction [cpdf_set_current_page\(\)](#) vous permet de quitter temporairement une page, et d'en modifier une autre.

La plus part des fonctions sont très simples d'emploi. Le plus difficile est probablement de créer un document PDF simple. L'exemple suivant devrait vous aider à démarrer. La page contient du texte qui utilise la police "Times–Roman" en taille 30, outlined. Le texte est souligné.

Exemple simple ClibPDF

```
<?php
$cpdf = cpdf_open(0);
cpdf_page_init($cpdf, 1, 0, 595, 842);
cpdf_add_outline($cpdf, 0, 0, 0, 1, "Page 1");
cpdf_set_font($cpdf, "Times–Roman", 30, "WinAnsiEncoding");
cpdf_set_text_rendering($cpdf, 1);
cpdf_text($cpdf, "Times Roman outlined", 50, 750);
cpdf_moveto($cpdf, 50, 740);
cpdf_lineto($cpdf, 330, 740);
cpdf_stroke($cpdf);
cpdf_finalize($cpdf);
Header("Content-type: application/pdf");
cpdf_output_buffer($cpdf);
cpdf_close($cpdf);
?>
```

La distribution `pdflib` contient un exemple plus complet, qui crée des séries de pages avec une horloge. Voici cet exemple converti en script PHP qui utilise l'extension ClibPDF :

Exemple pdfclock de la distribution pdflib 2.0

```

<?php
$radius = 200;
$margin = 20;
$pagecount = 40;
$pdf = cpdf_open(0);
cpdf_set_creator($pdf, "pdf_clock.php3");
cpdf_set_title($pdf, "Analog Clock");
while($pagecount-- > 0) {
    cpdf_page_init($pdf, $pagecount+1, 0, 2 * ($radius + $margin), 2 * ($radius + $margin), 1.0);
    cpdf_set_page_animation($pdf, 4, 0.5, 0, 0, 0); /* wipe */
    cpdf_translate($pdf, $radius + $margin, $radius + $margin);
    cpdf_save($pdf);
    cpdf_setrgbcolor($pdf, 0.0, 0.0, 1.0);
    /* minute strokes */
    cpdf_setlinewidth($pdf, 2.0);
    for ($alpha = 0; $alpha < 360; $alpha += 6)
    {
        cpdf_rotate($pdf, 6.0);
        cpdf_moveto($pdf, $radius, 0.0);
        cpdf_lineto($pdf, $radius-$margin/3, 0.0);
        cpdf_stroke($pdf);
    }
    cpdf_restore($pdf);
    cpdf_save($pdf);
    /* 5 minute strokes */
    cpdf_setlinewidth($pdf, 3.0);
    for ($alpha = 0; $alpha < 360; $alpha += 30)
    {
        cpdf_rotate($pdf, 30.0);
        cpdf_moveto($pdf, $radius, 0.0);
        cpdf_lineto($pdf, $radius-$margin, 0.0);
        cpdf_stroke($pdf);
    }
    $ltime = getdate();
    /* draw hour hand */
    cpdf_save($pdf);
    cpdf_rotate($pdf, -((($ltime['minutes']/60.0) + $ltime['hours'] - 3.0) * 30.0));
    cpdf_moveto($pdf, -$radius/10, -$radius/20);
    cpdf_lineto($pdf, $radius/2, 0.0);
    cpdf_lineto($pdf, -$radius/10, $radius/20);
    cpdf_closepath($pdf);
    cpdf_fill($pdf);
    cpdf_restore($pdf);
    /* draw minute hand */
    cpdf_save($pdf);
    cpdf_rotate($pdf, -((($ltime['seconds']/60.0) + $ltime['minutes'] - 15.0) * 6.0));
    cpdf_moveto($pdf, -$radius/10, -$radius/20);
    cpdf_lineto($pdf, $radius * 0.8, 0.0);
    cpdf_lineto($pdf, -$radius/10, $radius/20);
    cpdf_closepath($pdf);
    cpdf_fill($pdf);
    cpdf_restore($pdf);
    /* draw second hand */
    cpdf_setrgbcolor($pdf, 1.0, 0.0, 0.0);
    cpdf_setlinewidth($pdf, 2);
    cpdf_save($pdf);
    cpdf_rotate($pdf, -((($ltime['seconds'] - 15.0) * 6.0));
    cpdf_moveto($pdf, -$radius/5, 0.0);
    cpdf_lineto($pdf, $radius, 0.0);
    cpdf_stroke($pdf);
}

```

```

cpdf_restore($pdf);
/* draw little circle at center */
cpdf_circle($pdf, 0, 0, $radius/30);
cpdf_fill($pdf);
cpdf_restore($pdf);
cpdf_finalize_page($pdf, $pagecount+1);
}
cpdf_finalize($pdf);
Header("Content-type: application/pdf");
cpdf_output_buffer($pdf);
cpdf_close($pdf);
?>

```

10.9.1 cpdf_global_set_document_limits

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_global_set_document_limits](#)(int *maxpages*, int *maxfonts*, int *maximages*, int *maxannotations*, int *maxobjects*)

[PHP 4 >= 4.0b4]

[cpdf_global_set_document_limits\(\)](#) permet de fixer plusieurs limites au document PDF. Cette fonction doit être appelé avant [cpdf_open\(\)](#) pour être effective. Elle fixe les limites de tous les documents ouverts après. Voir aussi [cpdf_open\(\)](#).

10.9.2 cpdf_set_creator

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_creator](#)(string *creator*)

[PHP 3 >= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_creator\(\)](#) fixe le créateur d'un document PDF. Voir aussi [cpdf_set_subject\(\)](#), [cpdf_set_title\(\)](#), [cpdf_set_keywords\(\)](#).

10.9.3 cpdf_set_title

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_title](#)(string *title*)

[PHP 3 >= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_title\(\)](#) fixe le titre d'un document PDF. Voir aussi [cpdf_set_subject\(\)](#), [cpdf_set_creator\(\)](#), [cpdf_set_keywords\(\)](#).

10.9.4 cpdf_set_subject

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_subject](#)(string *subject*)

[PHP 3 >= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_subject\(\)](#) fixe le sujet d'un document PDF. Voir aussi [cpdf_set_title\(\)](#), [cpdf_set_creator\(\)](#), [cpdf_set_keywords\(\)](#).

10.9.5 cpdf_set_keywords

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_keywords](#)(string *keywords*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_keywords\(\)](#) fixe les mot clés d'un document PDF.

Voir aussi [cpdf_set_title\(\)](#), [cpdf_set_creator\(\)](#), [cpdf_set_subject\(\)](#).

10.9.6 cpdf_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [cpdf_open](#)(int *compression*, string *filename*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_open\(\)](#) ouvre un nouveau document PDF. Le premier paramètre *compression* active ou pas la compression, suivant qu'il vaut 0 ou 1. Le deuxième paramètre, optionnel, choisit le fichier de destination du document. Si il est omis, le document sera écrit en mémoire, et pourra être écrit dans un fichier avec [cpdf_save_to_file\(\)](#) ou envoyé à l'affichage avec [cpdf_output_buffer\(\)](#). Note : *La valeur retournée sera nécessaire pour les autres fonctions de ClibPDF comme premier paramètre.*

La librairie ClibPDF prend le nom de fichier "-" comme synonyme de stdout. Si PHP est compilé comme un module apache, cela ne fonctionnera pas, car la méthode d'envoi des données de ClibPDF ne fonctionne pas avec Apache. Vous pouvez résoudre ce problème en ne fournissant pas de nom de fichier, et en utilisant la fonction [cpdf_output_buffer\(\)](#) pour afficher le document PDF.

Voir aussi [cpdf_close\(\)](#), [cpdf_output_buffer\(\)](#).

10.9.7 cpdf_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_close](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_close\(\)](#) ferme un fichier PDF. Ce doit être la dernière fonction appelée, et elle apparaît même après [cpdf_finalize\(\)](#), [cpdf_output_buffer\(\)](#) et [cpdf_save_to_file\(\)](#).

Voir aussi [cpdf_open\(\)](#).

10.9.8 cpdf_page_init

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_page_init](#)(int *pdf document*, int *page number*, int *orientation*, double *height*, double *width*, double *unit*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_page_init\(\)](#) commence une nouvelle page, avec la hauteur *height* et la largeur *width*. La page a le numéro *page number* et l'orientation *orientation*. *orientation* vaut 0 pour portrait et 1 pour paysage. Le dernier paramètre, optionnel, *unit*, fixe l'unité pour le système de coordonnées. Cette valeur doit être un

nombre de points postscript, par unité. Etant donné que un pouce (inch) vaut 72 points, une valeur de 72 vaudra un pouce (inch). Par défaut, cette valeur vaut 72.

Voir aussi [cpdf_set_current_page\(\)](#).

10.9.9 cpdf_finalize_page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_finalize_page](#)(int *pdf document*, int *page number*)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

[cpdf_finalize_page\(\)](#) termine la page de numéro *page number*. Cette fonction fait que qu'une sauvegarde mémoire. Les pages terminées prennent moins de place, mais ne peuvent plus être modifiées.

Voir aussi [cpdf_page_init\(\)](#).

10.9.10 cpdf_finalize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_finalize](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_finalize\(\)](#) termine un document. Vous devez toujours appeler [cpdf_close\(\)](#) après.

Voir aussi [cpdf_close\(\)](#).

10.9.11 cpdf_output_buffer

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_output_buffer](#)(int *pdf document*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_output_buffer\(\)](#) envoie le document PDF dans un buffer mémoire de stdout. Le document doit avoir été créé en mémoire, ce qui est le cas si [cpdf_open\(\)](#) a été appelée dans paramètre de nom de fichier.

Voir aussi [cpdf_open\(\)](#).

10.9.12 cpdf_save_to_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_save_to_file](#)(int *pdf document*, string *filename*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_save_to_file\(\)](#) 'crit un document PDF dans un fichier, s'il a été créé en mémoire. Cette fonction n'est pas nécessaire si un nom de fichier a été fourni lors de l'appel à [cpdf_open\(\)](#).

Voir aussi [cpdf_output_buffer\(\)](#), [cpdf_open\(\)](#).

10.9.13 cpdf_set_current_page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_current_page](#)(int *pdf document*, int *page number*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_set_current_page\(\)](#) fixe la page courante, où toutes les prochaines opérations vont avoir lieu. On peut changer de page jusqu'à ce qu'une page soit terminée avec [cpdf_finalize_page\(\)](#). Voir aussi [cpdf_finalize_page\(\)](#).

10.9.14 cpdf_begin_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_begin_text](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_begin_text\(\)](#) démarre une section de texte. Elle doit être terminée avec [cpdf_end_text\(\)](#).

Affichage de texte

```
<?php
cpdf_begin_text($pdf);
cpdf_set_font($pdf, 16, "Helvetica", "WinAnsiEncoding");
cpdf_text($pdf, 100, 100, "Some text");
cpdf_end_text($pdf)
?>
```

Voir aussi [cpdf_end_text\(\)](#).

10.9.15 cpdf_end_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_end_text](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_end_text\(\)](#) termine une section de texte, commencée avec [cpdf_begin_text\(\)](#).

Affichage de texte

```
<?php
cpdf_begin_text($pdf);
cpdf_set_font($pdf, 16, "Helvetica", "WinAnsiEncoding");
cpdf_text($pdf, 100, 100, "Some text");
cpdf_end_text($pdf)
?>
```

Voir aussi [cpdf_begin_text\(\)](#).

10.9.16 cpdf_show

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_show](#)(int *pdf document*, string *text*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_show\(\)](#) imprime la chaîne *text*, à la position courante.

Voir aussi [cpdf_text\(\)](#), [cpdf_begin_text\(\)](#), [cpdf_end_text\(\)](#).

10.9.17 cpdf_show_xy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_show_xy](#) (int *pdf document*, string *text*, double *x-coor*, double *y-coor*, int *mode*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_show_xy\(\)](#) imprime la chaîne *text*, à la position de coordonnées (*x-koor*, *y-koor*). Le dernier paramètre optionnel est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé. Note : The function [cpdf_show_xy\(\)](#) est identique à [cpdf_text\(\)](#) sans les options.

Voir aussi [cpdf_text\(\)](#).

10.9.18 cpdf_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_text](#) (int *pdf document*, string *text*, double *x-coor*, double *y-coor*, int *mode*, double *orientation* , int *alignmode*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_text\(\)](#) imprime le text *text* à la position de coordonnées (*x-koor*, *y-koor*). Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé. Le paramètre optionnel *orientation* est un angle de rotation du texte, en degrés. Le paramètre optionnel *alignmode* détermine l'alignement du texte. Reportez vous à la doc de ClibPDF, pour les valeurs possibles. Voir aussi [cpdf_show_xy\(\)](#).

10.9.19 cpdf_set_font

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_font](#) (int *pdf document*, string *font name*, double *size*, string *encoding*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_font\(\)](#) selectionne la police courante, sa taille et l'encodage. Actuellement, seules les polices postscript sont supportées.

Le dernier paramètre *encoding* peut prendre les valeurs suivantes : "MacRomanEncoding", "MacExpertEncoding", "WinAnsiEncoding", et "NULL". "NULL" signifie qu'il faut utiliser l'encodage par défaut.

Reportez vous à la doc de ClibPDF, pour plus d'informations, notamment sur les polices asiatiques.

10.9.20 cpdf_set_leading

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_leading](#) (int *pdf document*, double *distance*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_leading\(\)](#) fixe la distance entre deux lignes. Cela servira si le texte est affiché par [cpdf_continue_text\(\)](#).
Voir aussi [cpdf_continue_text\(\)](#).

10.9.21 [cpdf_set_text_rendering](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_rendering](#) (int *pdf document*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_text_rendering\(\)](#) détermine le rendu du texte.

Les valeurs possibles pour *mode* sont : 0=texte plein, 1=texte stroke, 2=texte plein et stroke, 3=invisible, 4=texte plein et ajouté au chemin, 5=texte stroke et ajouté au chemin, 6=texte plein et stroke et ajouté au chemin, 7=et ajouté au chemin.

10.9.22 [cpdf_set_horiz_scaling](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_horiz_scaling](#) (int *pdf document*, double *scale*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_horiz_scaling\(\)](#) fixe l'échelle horizontale du texte à *scale* %.

10.9.23 [cpdf_set_text_rise](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_rise](#) (int *pdf document*, double *value*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_text_rise\(\)](#) fixe l'élévation du texte à *value* unités.

10.9.24 [cpdf_set_text_matrix](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_matrix](#) (int *pdf document*, array *matrix*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_text_matrix\(\)](#) fixe la matrice du texte, qui décrit la transformation appliquée à police.

10.9.25 [cpdf_set_text_pos](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_text_pos](#) (int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_text_pos\(\)](#) Fixe la position du texte pour le prochain appel à [cpdf_show\(\)](#).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par

défaut (72) qui est utilisé.

Voir aussi [cpdf_show\(\)](#), [cpdf_text\(\)](#).

10.9.26 cpdf_set_char_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_char_spacing](#)(int *pdf document*, double *space*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_char_spacing\(\)](#) fixe l'espacement des caractères.

Voir aussi [cpdf_set_word_spacing\(\)](#), [cpdf_set_leading\(\)](#).

10.9.27 cpdf_set_word_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_word_spacing](#)(int *pdf document*, double *space*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_set_word_spacing\(\)](#) fixe l'espacement des caractères.

Voir aussi [cpdf_set_char_spacing\(\)](#), [cpdf_set_leading\(\)](#).

10.9.28 cpdf_continue_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_continue_text](#)(int *pdf document*, string *text*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_continue_text\(\)](#) imprime le texte *text* à la ligne suivante.

Voir aussi [cpdf_show_xy\(\)](#), [cpdf_text\(\)](#), [cpdf_set_leading\(\)](#), [cpdf_set_text_pos\(\)](#).

10.9.29 cpdf_stringwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [cpdf_stringwidth](#)(int *pdf document*, string *text*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_stringwidth\(\)](#) retourne la taille de la chaîne *text*. Une police doit avoir déjà été choisie.

Voir aussi [cpdf_set_font\(\)](#).

10.9.30 cpdf_save

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_save](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_save\(\)](#) sauve l'environnement courant. Cette fonction est similaire à la commande postscript gsave. Très pratique quand vous devez faire des translations et rotations sur un objet, mais sans affecter les autres.

Voir aussi [cpdf_restore\(\)](#).

[10.9.31 cpdf_restore](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_restore](#) (int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_restore\(\)](#) restaure l'environnement sauvé par [cpdf_save\(\)](#). Cette fonction est similaire à la commande postscript grestore. Très pratique quand vous devez faire des translations et rotations sur un objet, mais sans affecter les autres.

Sauver/Restaurer

```
<?php
cpdf_save($pdf);
// plein de transformations, translations, ...
cpdf_restore($pdf)
?>
```

Voir aussi [cpdf_save\(\)](#).

[10.9.32 cpdf_translate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_translate](#) (int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_translate\(\)](#) modifie l'origine du système de coordonnées en plaçant l'origine aux coordonnées (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. S'il prend la valeur de 0 (ou s'il est omis), c'est la valeur par défaut (72) qui est utilisé.

[10.9.33 cpdf_scale](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_scale](#) (int *pdf document*, double *x-scale*, double *y-scale*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_scale\(\)](#) modifie l'échelle dans les deux directions.

[10.9.34 cpdf_rotate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rotate](#) (int *pdf document*, double *angle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_rotate\(\)](#) effectue une rotation, d'un angle de *angle* degrés.

10.9.35 cpdf_setflat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setflat](#) (int *pdf document*, double *value*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setflat\(\)](#) fixe la platitude (flatness), entre 0 et 100.

10.9.36 cpdf_setlinejoin

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setlinejoin](#) (int *pdf document*, long *value*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setlinejoin\(\)](#) fixe le paramètre linejoin à une valeur *value*, entre 0 et 2. 0 = miter, 1 = round, 2 = bevel.

10.9.37 cpdf_setlinecap

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setlinecap](#) (int *pdf document*, int *value*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setlinecap\(\)](#) fixe le paramètre linecap à une valeur *value* entre 0 et 2. 0 = butt end, 1 = round, 2 = projecting square.

10.9.38 cpdf_setmiterlimit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setmiterlimit](#) (int *pdf document*, double *value*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setmiterlimit\(\)](#) fixe le paramètre "miter limit" à une valeur supérieure ou égale à 1.

10.9.39 cpdf_setlinewidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setlinewidth](#) (int *pdf document*, double *width*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setlinewidth\(\)](#) fixe la largeur de ligne à la valeur de *width*.

10.9.40 cpdf_setdash

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setdash](#) (int *pdf document*, double *white*, double *black*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setdash\(\)](#) fixe le motif de pointillé à **white** unité de blanc et **black** unités de noir. Si les deux sont à 0, une ligne pleine est affichée.

10.9.41 cpdf_newpath

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_newpath](#)(int *pdf_document*)
[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_newpath\(\)](#) commence un nouveau chemin dans le document *pdf_document*.

10.9.42 cpdf_moveto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_moveto](#)(int *pdf_document*, double *x-koor*, double *y-koor*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_moveto\(\)](#) fixe le point courant aux coordonnées (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

10.9.43 cpdf_rmoveto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rmoveto](#)(int *pdf_document*, double *x-koor*, double *y-koor*, int *mode*)
[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_rmoveto\(\)](#) fixe le point courant aux coordonnées (*x-koor*, *y-koor*), relativement.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_moveto\(\)](#).

10.9.44 cpdf_curveto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_curveto](#)(int *pdf_document*, double *x1*, double *y1*, double *x2*, double *y2*, double *x3*, double *y3*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_curveto\(\)](#) dessine une courbe de Bezier, entre le point courant et le point (*x3*, *y3*), en utilisant les points de contrôle (*x1*, *y1*) et (*x2*, *y2*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_moveto\(\)](#), [cpdf_rmoveto\(\)](#), [cpdf_rlineto\(\)](#), [cpdf_lineto\(\)](#).

10.9.45 [cpdf_lineto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_lineto](#) (int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_lineto\(\)](#) dessine une ligne entre le point courant et le point de coordonnées (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_moveto\(\)](#), [cpdf_rmoveto\(\)](#), [cpdf_curveto\(\)](#).

10.9.46 [cpdf_rlineto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rlineto](#) (int *pdf document*, double *x-koor*, double *y-koor*, int *mode*)
[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_rlineto\(\)](#) dessine une ligne entre le point courant et le point de coordonnées relatives (*x-koor*, *y-koor*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_moveto\(\)](#), [cpdf_rmoveto\(\)](#), [cpdf_curveto\(\)](#).

10.9.47 [cpdf_circle](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_circle](#) (int *pdf document*, double *x-koor*, double *y-koor*, double *radius*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_circle\(\)](#) dessine un cercle de centre (*x-koor*, *y-koor*) et de rayon *radius*.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_arc\(\)](#).

10.9.48 [cpdf_arc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_arc](#) (int *pdf document*, double *x-koor*, double *y-koor*, double *radius*, double *start*, double *end*, int *mode*)
[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_arc\(\)](#) Dessine un arc de cercle, dont le centre est au point (*x-koor*, *y-koor*) et l'angle est *radius*, commençant à l'angle *start* et finissant à l'angle *end*.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_circle\(\)](#).

10.9.49 cpdf_rect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_rect](#)(int *pdf document*, double *x-koor*, double *y-koor*, double *width*, double *height*, int *mode*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_rect\(\)](#) dessine un rectangle dont le coin inférieur droit est au point (*x-koor*, *y-koor*). La largeur est *width*. La hauteur est *height*.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

10.9.50 cpdf_closepath

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_closepath](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_closepath\(\)](#) ferme le chemin courant.

10.9.51 cpdf_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_stroke](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_stroke\(\)](#) dessine une ligne le long du chemin.

Voir aussi [cpdf_closepath\(\)](#), [cpdf_closepath_stroke\(\)](#).

10.9.52 cpdf_closepath_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_closepath_stroke](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_closepath_stroke\(\)](#) est une combinaison de [cpdf_closepath\(\)](#) et [cpdf_stroke\(\)](#).

Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#).

10.9.53 cpdf_fill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_fill](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_fill\(\)](#) remplit l'intérieur du chemin courant avec la couleur courante.

Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#), [cpdf_setgray_fill\(\)](#), [cpdf_setgray\(\)](#), [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.54 [cpdf_fill_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_fill_stroke](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_fill_stroke\(\)](#) remplit l'intérieur du chemin avec la couleur courante, et dessine le chemin.

Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#), [cpdf_fill\(\)](#), [cpdf_setgray_fill\(\)](#), [cpdf_setgray\(\)](#), [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.55 [cpdf_closepath_fill_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_closepath_fill_stroke](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_closepath_fill_stroke\(\)](#) remplit le chemin, dessine le bord et ferme le chemin.

Voir aussi [cpdf_closepath\(\)](#), [cpdf_stroke\(\)](#), [cpdf_fill\(\)](#), [cpdf_setgray_fill\(\)](#), [cpdf_setgray\(\)](#), [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.56 [cpdf_clip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_clip](#)(int *pdf document*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_clip\(\)](#) aligne les dessins sur le chemin courant.

10.9.57 [cpdf_setgray_fill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setgray_fill](#)(int *pdf document*, double *value*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setgray_fill\(\)](#) remplace le niveau de gris, couleur de remplissage courante, par *value*.

Voir aussi [cpdf_setrgbcolor_fill\(\)](#).

10.9.58 [cpdf_setgray_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setgray_stroke](#)(int *pdf document*, double *gray value*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setgray_stroke\(\)](#) remplace le niveau de gris, couleur de dessin courante, par *value*.

Voir aussi [cpdf_setrgbcolor_stroke\(\)](#).

10.9.59 cpdf_setgray

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setgray](#) (int *pdf document*, double *gray value*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setgray_stroke\(\)](#) remplace le niveau de gris, couleur de dessin et de remplissage, par *value*.

Voir aussi [cpdf_setrgbcolor_stroke\(\)](#), [cpdf_setrgbcolor_fill\(\)](#).

10.9.60 cpdf_setrgbcolor_fill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setrgbcolor_fill](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setrgbcolor_fill\(\)](#) remplace la couleur de remplissage, par la couleur rgb (*red value*, *green value*, *blue value*).

Voir aussi [cpdf_setrgbcolor_stroke\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.61 cpdf_setrgbcolor_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setrgbcolor_stroke](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setrgbcolor_stroke\(\)](#) remplace la couleur de dessin, par la couleur rgb (*red value*, *green value*, *blue value*).

Voir aussi [cpdf_setrgbcolor_fill\(\)](#), [cpdf_setrgbcolor\(\)](#).

10.9.62 cpdf_setrgbcolor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_setrgbcolor](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b4]

[cpdf_setrgbcolor_stroke\(\)](#) remplace la couleur de remplissage et de dessin, par la couleur rgb (*red value*, *green value*, *blue value*).

Voir aussi [cpdf_setrgbcolor_stroke\(\)](#), [cpdf_setrgbcolor_fill\(\)](#).

10.9.63 cpdf_add_outline

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_add_outline](#) (int *pdf document*, string *text*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_add_outline\(\)](#) ajoute un signet à la page courante, avec le texte *text* qui pointe sur la page courante.

Ajouter une mise en relief

```
<?php
$cpdf = cpdf_open(0);
cpdf_page_init($cpdf, 1, 0, 595, 842);
cpdf_add_outline($cpdf, 0, 0, 0, 1, "Page 1");
// ...
// quelques dessins
// ...
cpdf_finalize($cpdf);
Header("Content-type: application/pdf");
cpdf_output_buffer($cpdf);
cpdf_close($cpdf);
?>
```

10.9.64 cpdf_set_page_animation

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_set_page_animation](#) (int *pdf document*, int *transition*, double *duration*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_set_page_animation\(\)](#) fixe l'animation de la transition entre les pages.

La valeur du paramètre de transition *transition* peut être :

- 0 pour aucune,
- 1 pour deux lignes en travers de l'écran, qui révèlent la prochaine page,
- 2 pour plusieurs lignes en travers de l'écran, qui révèlent la prochaine page,
- 3 pour une boîte qui révèle la prochaine page,
- 4 pour une seule ligne en travers de l'écran, qui révèle la prochaine page,
- 5 pour l'ancienne page qui se dissout
- 6 pour un effet de dissolution d'un angle à l'autre
- 7 pour le remplacement simple (par défaut)

La valeur de *duration* est le nombre de secondes avant le passage à la page suivante.

10.9.65 cpdf_import_jpeg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [cpdf_import_jpeg](#) (int *pdf document*, string *file name*, double *x-koor*, double *y-koor*, double *angle*, double *width*, double *height*, double *x-scale*, double *y-scale*, int *mode*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_import_jpeg\(\)](#) ouvre une image JPG, enregistré dans le fichier *file name*. Le format de l'image doit être JPEG. L'image est placée dans la page courante, aux coordonnées (*x-koor*, *y-koor*). L'image subira une

rotation d'un angle de *angle* degrés.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisée.

Voir aussi [cpdf_place_inline_image\(\)](#).

10.9.66 cpdf_place_inline_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_place_inline_image](#)(int *pdf document*, int *image*, double *x-koor*, double *y-koor*, double *angle*, double *width*, double *height*, int *mode*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0b4]

[cpdf_place_inline_image\(\)](#) places une image créée par un script PHP, dans la page, à la position (*x-koor*, *y-koor*). L'image peut être mise à l'échelle, en même temps.

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

Voir aussi [cpdf_import_jpeg\(\)](#).

10.9.67 cpdf_add_annotation

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [cpdf_add_annotation](#)(int *pdf document*, double *llx*, double *lly*, double *urx*, double *ury*, string *title*, string *content*, int *mode*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b4]

[cpdf_add_annotation\(\)](#) ajoute une note, dont le coin inférieur droit est (*llx*, *lly*) et le coin supérieur droit est (*urx*, *ury*).

Le paramètre *mode* est une unité de longueur. Si il prend la valeur de 0 (ou si il est omis), c'est la valeur par défaut (72) qui est utilisé.

10.10 CURL

[\[Notes en ligne\]](#)

PHP supporte libcurl, une librairie créée par Daniel Stenberg, qui vous permet de vous connecter de communiquer avec de nombreux serveurs, grâce à de nombreux protocoles. libcurl supporte actuellement les protocoles suivants : http, https, ftp, gopher, telnet, dict, file, et ldap. libcurl supporte aussi les certificats HTTPS, les POST HTTP, PUT HTTP, le chargement par FTP (ce qui peut être fait par l'extension FTP), les chargement par formulaire HTTP, les proxies, les cookies et l'authentification par mot de passe et nom de compte.

Pour pouvoir utiliser les fonctions CURL, vous devez installer le package [CURL](#). PHP requiert la version CURL 7.0.2-beta ou plus récente. PHP ne fonctionnera pas avec une version inférieure à la version 7.0.2-beta.

Pour utiliser CURL depuis les scripts PHP, vous devez aussi compiler PHP avec l'option `--with-curl[=DIR]` où DIR est le chemin jusqu'au dossier contenant les dossier `'lib'` et `'include'`. Dans le dossier `'include'` il doit se trouver un dossier appelé `'curl'`, qui contient notamment les fichiers `'easy.h'` et `'curl.h'`. Il doit aussi se trouver un fichier nommé `'libcurl.a'` dans le dossier `'lib'`.

Une fois que vous avez compilé PHP avec le support CURL, vous pouvez commencer à l'exploiter avec vos scripts PHP. Le principe de fonctionnement est d'initialiser une session CURL avec [curl_init\(\)](#), puis de

choisir toutes vos options de transfert avec [curl_exec\(\)](#) et de finir votre session avec [curl_close\(\)](#). Voici un exemple d'utilisation des fonctions CURL, qui récupère la page principale de PHP :

Utilisation de CURL et PHP pour récupérer une page

```
<?php
$ch = curl_init ("http://www.php.net/");
$fp = fopen ("php_homepage.txt", "w");
curl_setopt ($ch, CURLOPT_INFILE, $fp);
curl_setopt ($ch, CURLOPT_HEADER, 0);
curl_exec ($ch);
curl_close ($ch);
fclose ($fp);
?>
```

10.10.1 curl_init

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [curl_init](#) (string *url*)

[PHP 4 >= 4.0.2]

[curl_init\(\)](#) initialise une nouvelle session et retourne un identifiant de session CURL, à utiliser avec les fonctions [curl_setopt\(\)](#), [curl_exec\(\)](#) et [curl_close\(\)](#). Si le paramètre optionnel *url* est fourni, alors CURLOPT_URL prendra cette valeur. Vous pouvez manuellement fixer cette valeur avec la fonction [curl_setopt\(\)](#).

Initialiser une sessions CURL et récupération d'une page web.

```
<?php
$ch = curl_init();
curl_setopt ($ch, CURLOPT_URL, "http://www.zend.com/");
curl_setopt ($ch, CURLOPT_HEADER, 0);
curl_exec ($ch);
curl_close ($ch);
?>
```

Voir aussi : [curl_close\(\)](#), [curl_setopt\(\)](#).

10.10.2 curl_setopt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [curl_setopt](#) (int *ch* , string *option* , mixed *value*)

[PHP 4 >= 4.0.2]

[curl_setopt\(\)](#) fixe les options de transfert de la session CURL identifiée par *ch*. *option* est le nom de l'option à fixer, et *value* est sa valeur.

value doit être de type "long" pour les options suivantes (spécifiée par *option*) :

- **CURLOPT_INFILESIZE**: Lorsque vous téléchargez un fichier sur un site distant, cette option sert à indiquer à PHP la taille maximale du fichier attendu.
- **CURLOPT_VERBOSE**: Choisissez une valeur non nulle pour que CURL vous affiche tous les

événements.

- ***CURLOPT_HEADER***: Choisissez une valeur non nulle pour que CURL inclus l'entête dans la valeur de retour.
- ***CURLOPT_NOPROGRESS***: Choisissez une valeur non nulle pour que PHP n'affiche pas l'état des transferts CURL. Note : *PHP choisit automatiquement une valeur non nulle. Ne changez cette valeur que le temps du débogage.*
- ***CURLOPT_NOBODY***: Choisissez une valeur non nulle pour que le corps du transfert ne soit pas inclus dans la valeur de retour.
- ***CURLOPT_FAILONERROR***: Choisissez une valeur non nulle pour que PHP traite silencieusement les codes HTTP supérieurs à 300. Le comportement par défaut est de retourner la page normalement, en ignorant ce code.
- ***CURLOPT_UPLOAD***: Choisissez une valeur non nulle pour que PHP prépare un chargement.
- ***CURLOPT_POST***: Choisissez une valeur non nulle pour que PHP fasse un HTTP POST. Un POST est un encodage normal "application/x-www-form-urlencoded", utilisé couramment par les formulaires HTML.
- ***CURLOPT_FTPLISTONLY***: Choisissez une valeur non nulle pour que PHP ne fasse que lister les noms d'un dossier FTP.
- ***CURLOPT_FTPAPPEND***: Choisissez une valeur non nulle pour que PHP concatène le fichier distant, plutôt que de l'écraser.
- ***CURLOPT_NETRC***: Choisissez une valeur non nulle pour que PHP scanne votre fichier ~/.netrc et utilise votre nom de compte et mot de passe sur le site distant que vous souhaitez contacter.
- ***CURLOPT_FOLLOWLOCATION***: Choisissez une valeur non nulle pour suivre toutes les entêtes "Location: " que le serveur envoie dans les entêtes HTTP (notez que cette fonction est récursive, et que PHP suivra toutes les entêtes "Location: " qu'il trouvera).
- ***CURLOPT_PUT***: Choisissez une valeur non nulle pour que pour chargement se fasse par HTTP PUT. Le fichier à charger doit être fixé avec les options CURLOPT_INFILE et CURLOPT_INFILESIZE.
- ***CURLOPT_MUTE***: Choisissez une valeur non nulle pour que PHP soit totalement silencieux concernant toutes les fonctions CURL.
- ***CURLOPT_TIMEOUT***: Passez un entier "long" comme paramètre qui représente le temps maximum d'exécution de la fonction CURL.
- ***CURLOPT_LOW_SPEED_LIMIT***: Passez un entier long qui représente la vitesse minimale en octets par secondes en dessous de laquelle, et pendant CURLOPT_LOW_SPEED secondes, PHP considèrera qu'elle est trop lente, et annulera le transfert.
- ***CURLOPT_LOW_SPEED_TIME***: Passez un entier "long" qui représente le temps en secondes, qui, si la vitesse de transfert reste en dessous de CURLOPT_LOW_SPEED_LIMIT, PHP considèrera que la connexion est trop lente, et l'annulera.
- ***CURLOPT_RESUME_FROM***: Passez un entier "long", qui représente l'offset, en octets, à partir duquel vous voulez commencer le transfert.
- ***CURLOPT_SSLVERSION***: Passez un entier "long" qui contient la version de SSL (2 ou 3) à utiliser. Par défaut, PHP essaiera de le déterminer par lui-même, bien que dans certains cas, il vous faudra le faire manuellement.
- ***CURLOPT_TIMECONDITION***: Passez un entier "long" qui définit comment CURLOPT_TIMEVALUE est utilisé. Vous pouvez choisir entre les valeurs TIMECOND_IFMODSINCE ou TIMECOND_ISUNMODSINCE. C'est une fonctionnalité HTTP.
- ***CURLOPT_TIMEVALUE***: Passez un entier "long" qui représente le temps en secondes depuis le 1er janvier 1970. Cette valeur sera utilisée comme spécifié dans l'option CURLOPT_TIMEVALUE. Par défaut, TIMECOND_IFMODSINCE sera utilisé.

value doit être une chaîne de caractères pour les valeurs suivantes de *option*

- ***CURLOPT_URL***: L'URL que PHP va récupérer. Vous pouvez aussi choisir cette valeur lors de l'appel à [curl_init\(\)](#) fonction.
- ***CURLOPT_USERPWD***: Passez une chaîne de caractères au format [nom]:[mot de passe], pour que PHP l'utilise lors de la connexion.
- ***CURLOPT_PROXYUSERPWD***: Passez une chaîne de caractères au format [nom]:[mot de passe], pour que PHP l'utilise lors de la connexion à un proxy HTTP.
- ***CURLOPT_RANGE***: Passez une chaîne de caractères qui représente la plage de valeur que vous désirez. Elle est au format "X-Y", où les valeurs de X ou Y peuvent être omises. Le transfert HTTP supporte aussi plusieurs intervalles, séparé par des virgules : X-Y,N-M.
- ***CURLOPT_POSTFIELDS***: Passez une chaîne de caractères qui contient toutes les données à passer lors d'une opération de HTTP POST.
- ***CURLOPT_REFERER***: Passez une chaîne de caractères qui contient l'entête de "REFERER", utilisé lors d'une requête HTTP.
- ***CURLOPT_USERAGENT***: Passez une chaîne de caractères qui contient l'entête "user-agent" utilisé dans une requête HTTP.
- ***CURLOPT_FTPPORT***: Passez une chaîne de caractères qui désignera l'adresse IP utilisée pour l'instruction FTP "PORT". L'instruction POST indique au serveur distant de se connecter cette adresse IP. La chaîne peut être une adresse IP, un nom d'hôte, un nom d'interface réseau (sous UNIX), ou juste '-', pour utiliser les IP par défaut du système.
- ***CURLOPT_COOKIE***: Passez une chaîne de caractères qui contiendra le contenu du cookie, à transmettre dans l'entête HTTP.
- ***CURLOPT_SSLCERT***: Passez une chaîne de caractères qui contiendra le nom de fichier du certificat, au format PEM.
- ***CURLOPT_SSLCERTPASSWD***: Passez une chaîne de caractères qui contient le mot de passe nécessaire pour utiliser le certificat CURLOPT_SSLCERT.
- ***CURLOPT_COOKIEFILE***: Passez une chaîne de caractères qui contiendra le nom du fichier contenant les données de cookie. Le fichier de cookie peut être au format Netscape, ou simplement des entêtes HTTP écrites dans un fichier.
- ***CURLOPT_CUSTOMREQUEST***: Passez une chaîne de caractères qui sera utilisé à la place de GET ou HEAD lors des requêtes HTTP. Cette commande est pratique pour effectuer un DELETE, ou une autre commande HTTP exotique. Note : *N'utilisez pas cette commande sans vous assurer que le serveur l'accepte.*

Les options suivantes requièrent un pointeur de fichier, qui est obtenu avec la fonction [fopen\(\)](#) :

- ***CURLOPT_FILE***: Le fichier de sortie de votre transfert. Par défaut, STDOUT.
- ***CURLOPT_INFILE***: Le fichier d'entrée de votre transfert.
- ***CURLOPT_WRITEHEADER***: Le fichier de destination de l'entête de la sortie du transfert.
- ***CURLOPT_STDERR***: Le fichier d'erreurs.

10.10.3 curl_exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [curl_exec](#) (int *ch*)

[PHP 4 >= 4.0.2]

Cette fonction doit être appelée après l'initialisation et le paramétrage d'une session CURL. Son but est simplement d'exécuter la session *ch*.

10.10.4 curl_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [curl_close](#) (int *ch*)

[PHP 4 >= 4.0.2]

Cette fonction ferme une session CURL et libère toutes les ressources réservées. L'identifiant CURL, *ch*, est aussi effacé.

10.10.5 curl_version

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [curl_version](#)

[PHP 4 >= 4.0.2]

[curl_version\(\)](#) retourne une chaîne avec la version courante de la librairie CURL.

10.11 Paiement Cybercash

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que si PHP a été compilé avec l'option `--with-cybercash=[DIR]`. Ces fonctions ont été ajoutées dans PHP 4.

10.11.1 cybercash_encr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [cybercash_encr](#) (string *wmk*, string *sk*, string *inbuff*)

[PHP 4 >= 4.0b4]

[cybercash_encr\(\)](#) retourne un tableau associatif, contenant les éléments "errcode" et, si "errcode" vaut FALSE, "outbuff" (string), "outLth" (long) et "macbuff" (string).

10.11.2 cybercash_decr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [cybercash_decr](#) (string *wmk*, string *sk*, string *inbuff*)

[PHP 4 >= 4.0b4]

[cybercash_decr\(\)](#) retourne un tableau associatif, contenant les éléments "errcode" et, si "errcode" vaut FALSE, "outbuff" (string), "outLth" (long) et "macbuff" (string).

10.11.3 cyberscash_base64_encode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [cyberscash_base64_encode](#) (string *inbuff*)
[PHP 4 >= 4.0b4]

10.11.4 cyberscash_base64_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [cyberscash_base64_decode](#) (string *inbuff*)
[PHP 4 >= 4.0b4]

10.12 Dates et heures

[\[Notes en ligne\]](#)

10.12.1 checkdate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [checkdate](#) (int *month*, int *day*, int *year*)
[PHP 3, PHP 4]

[checkdate\(\)](#) retourne TRUE si la date fournie est valide, et sinon FALSE. La date est considérée comme valide si :

- L'année est comprise entre 0 et 32767 inclus
- Le mois est compris entre 1 et 12 inclus
- Le jour est compris dans l'intervalle de date du mois. Les années bissextiles sont prises en compte.

10.12.2 date

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [date](#) (string *format*, int *timestamp*)
[PHP 3, PHP 4]

[date\(\)](#) retourne une date sous forme d'une chaîne, au format donné par la chaîne format. La date est fournie sous la forme d'un *timestamp*. Par défaut, la date courante est utilisée.

Les caractères suivants sont utilisés pour spécifier le format :

- a – "am" (matin) ou "pm" (après-midi)
- A – "AM" (matin) ou "PM" (après-midi)

- B – Heure Internet Swatch
- d – Jour du mois, sur deux chiffres (éventuellement avec un zéro) : "01" à "31"
- D – Jour de la semaine, en trois lettres (et en anglais) : par exemple "Fri" (pour Vendredi)
- F – Mois, textuel, version longue; en anglais, i.e. "January" (pour Janvier)
- g – Heure, au format 12h, sans les zéros initiaux i.e. "1" à "12"
- G – Heure, au format 24h, sans les zéros initiaux i.e. "0" à "23"
- h – Heure, au format 12h, "01" à "12"
- H – heure, au format 24h, "00" à "23"
- i – Minutes; "00" à "59"
- I (i majuscule) – "1" si l'heure d'hivers est activée, "0" sinon.
- j – Jour du mois sans les zéros initiaux: "1" à "31"
- l – ('L' minuscule) – Jour de la semaine, textuel, version longue; en anglais, i.e. "Friday" (pour Vendredi)
- L – Booléen pour savoir si l'année est bisextile ("1") ou pas ("0")
- m – Mois; i.e. "01" à "12"
- M – Mois, en trois lettres (et en anglais) : par exemple "Apr" (pour Avril)
- n – Mois sans les zéros initiaux; i.e. "1" à "12"
- r – Format de date RFC 822; i.e. "Thu, 21 Dec 2000 16:01:07 +0200"
- s – Secondes; i.e. "00" à "59"
- S – Suffixe ordinal d'un nombre, en anglais, sur deux lettres : i.e. "th", "nd"
- t – Nombre de jour dans le mois donné, i.e. "28" à "31"
- T – Fuseau horaire de la machine ; i.e. "MET"
- U – Secondes depuis une époque
- w – Jour de la semaine, numérique, i.e. "0" (Dimanche) to "6" (Samedi)
- Y – Année, 4 chiffres; i.e. "1999"
- y – Année, 2 chiffres; i.e. "99"
- z – Jour de l'année; i.e. "0" à "365"
- Z – Décalage horaire en secondes (i.e. "-43200" à "43200")

Les caractères non reconnus seront imprimés tels quel. "Z" retournera toujours "0" lorsqu'il est utilisé avec [gmdate\(\)](#).

Exemple avec date()

```
print (date("l dS of F Y h:i:s A"));
print ("July 1, 2000 is on a " . date("l", mktime(0,0,0,7,1,2000)));
```

Il est possible d'utiliser [date\(\)](#) et [mktime\(\)](#) ensemble pour générer des dates dans futur ou dans passé.

Exemples avec date() et mktime()

```
<?php
$tomorrow = mktime (0,0,0,date("m") ,date("d")+1,date("Y"));
$lastmonth = mktime (0,0,0,date("m")-1,date("d"), date("Y"));
$nextyear = mktime (0,0,0,date("m"), date("d"), date("Y")+1);
?>
```

Pour formater des dates dans d'autres langues, utilisez les fonctions [setlocale\(\)](#) et [strftime\(\)](#). Voir aussi [gmdate\(\)](#) et [mktime\(\)](#).

10.12.3 getdate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getdate](#)(int *timestamp*)
[PHP 3, PHP 4]

[getdate\(\)](#) retourne un tableau associatif contenant les informations de date et heure du timestamp *timestamp* (lorsqu'il est fourni), avec les champs suivants :

- "seconds" – secondes
- "minutes" – minutes
- "hours" – heures
- "mday" – jour du mois
- "wday" – jour de la semaine, numérique
- "mon" – mois, numérique
- "year" – année, numérique
- "yday" – jour de l'année, numérique; i.e. "299"
- "weekday" – jour de la semaine, texte complet (en anglais); i.e. "Friday"
- "month" – mois, texte complet (en anglais); i.e. "January"

10.12.4 gettimeofday

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gettimeofday](#)(void)
[PHP 3 >= 3.0.7, PHP 4 >= 4.0b4]

[gettimeofday\(\)](#) est une interface vers gettimeofday(2). Elle retourne un tableau associatif qui contient les informations retournées par le système :

- "sec" – secondes
- "usec" – microsecondes
- "minuteswest" – minutes de décalage par rapport à Greenwich, vers l'Ouest.
- "dstime" – type de correction dst

10.12.5 gmdate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gmdate](#)(string *format*, int *timestamp*)
[PHP 3, PHP 4]

[gmdate\(\)](#) est identique à la fonction [date\(\)](#), hormis le fait que le temps retourné est GMT (Greenwich Mean Time) Par exemple, en Finlande (GMT +0200), la première ligne ci-dessous affiche "Jan 01 1998 00:00:00", tandis que la seconde "Dec 31 1997 22:00:00".

Exemple avec gmdate()


```
<?php
echo date ("M d Y H:i:s", mktime (0,0,0,1,1,1998));
echo gmdate ("M d Y H:i:s", mktime (0,0,0,1,1,1998));
?>
```

Voir aussi [date\(\)](#), [mktime\(\)](#), et [gmmktime\(\)](#).

10.12.6 gmmktime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmmktime](#) (int *hour*, int *minute*, int *second*, int *month*, int *day*, int *year*, int *is_dst*)

[PHP 3, PHP 4]

Identique à [mktime\(\)](#) hormis le fait que les paramètres passés sont GMT.

10.12.7 gmstrftime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gmstrftime](#) (string *format*, int *timestamp*)

[PHP 3 >= 3.0.12, PHP 4 >= 4.0RC2]

[gmstrftime\(\)](#) se comporte exactement comme [strftime\(\)](#) hormis le fait l'heure utilisée est celle de Greenwich (Greenwich Mean Time (GMT)). Par exemple, dans la zone Eastern Standard Time (est des USA) (GMT -0500), la première ligne de l'exemple ci dessous affiche "Dec 31 1998 20:00:00", tandis que la seconde affiche "Jan 01 1999 01:00:00".

Exemple avec gmstrftime()

```
setlocale ('LC_TIME', 'en_US');
echo strftime ("%b %d %Y %H:%M:%S", mktime (20,0,0,12,31,98))."\n";
echo gmstrftime ("%b %d %Y %H:%M:%S", mktime (20,0,0,12,31,98))."\n";
```

Voir aussi [strftime\(\)](#).

10.12.8 localtime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [localtime](#) (int *timestamp* , bool *is_associative*)

[PHP 4 >= 4.0RC2]

[localtime\(\)](#) retourne un tableau identique à la structure retournée par la fonction C localtime. Le premier argument *timestamp* est un timestamp UNIX. S'il n'est pas fourni, l'heure courante est utilisée. Le second argument *is_associative*, s'il est mis à 0 ou ignoré, force [localtime\(\)](#) à retourner un tableau à index numérique. S'il est mis à 1, [localtime\(\)](#) retourne un tableau associatif, avec tous les éléments de la structure C, accessible avec les clés suivantes :

- "tm_sec" – secondes

- "tm_min" – minutes
- "tm_hour" – heure
- "tm_mday" – jour du mois
- "tm_mon" – mois de l'année
- "tm_year" – Année, incompatible an 2000
- "tm_wday" – Jour de la semaine
- "tm_yday" – Jour de l'année
- "tm_isdst" – Est ce que l'heure d'hiver a pris effet

10.12.9 microtime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [microtime](#) (void)

[PHP 3, PHP 4]

[microtime\(\)](#) retourne la chaîne "msec sec" avec sec qui est mesurée en secondes depuis le début de l'époque UNIX, (1er janvier 1970 00:00:00 GMT), et msec qui est le nombre de microsecondes de cette heure. Cette fonction est seulement disponible sur les systèmes d'exploitation qui supportent la fonction gettimeofday(). Voir aussi [time\(\)](#).

10.12.10 mktime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mktime](#) (int *hour*, int *minute*, int *second*, int *month*, int *day*, int *year*, int *is_dst*)

[PHP 3, PHP 4]

ATTENTION : l'ordre des arguments est différent de celui de la commande UNIX habituelle mktime(), et fournit des résultats aléatoires si on oublie cet ordre. C'est une erreur très commune que de se tromper de sens.

[mktime\(\)](#) retourne un timestamp UNIX correspondant aux arguments fournis. Ce timestamp est un entier long, contenant le nombre de secondes entre le début de l'époque UNIX (1er Janvier 1970) et le temps spécifié.

Les arguments peuvent être omis, de gauche à droite, et tous les arguments manquants sont utilisés avec la valeur courante de l'heure et du jour.

is_dst peut être mis à 1 si l'heure d'hiver est appliquée, 0 si elle ne l'est pas, et -1 (par défaut) si on ne sait pas.

Note : *is_dst* a été ajouté à partir de la version 3.0.10.

[mktime\(\)](#) est pratique pour faire des calculs de dates et des validations, car elle va automatiquement corriger les valeurs invalides. Par exemple, toutes les lignes suivantes vont retourner la même date : "Jan-01-1998".

Exemple mktime()

```
<?php
echo date ("M-d-Y", mktime (0,0,0,12,32,1997));
echo date ("M-d-Y", mktime (0,0,0,13,1,1997));
echo date ("M-d-Y", mktime (0,0,0,1,1,1998));
echo date ("M-d-Y", mktime (0,0,0,1,1,98));
?>
```

year peut prendre deux ou quatre chiffres, avec les valeurs entre 0–69 qui correspondent à 2000–2069 et 70–99 à 1970–1999 (sur les systèmes où `time_t` sont sur des entiers 32bit signés, comme cela se fait le plus souvent de nos jours, **year** est valide dans l'intervalle 1902 et 2037).

Le dernier jours d'un mois peut être décrit comme le jour "0" du mois suivant, et non pas le jour –1. Les deux exemples suivants vont donner : "Le dernier jour de Février 2000 est: 29".

Dernier jour du mois

```
<?php
$lastday = mktime (0,0,0,3,0,2000);
echo strftime ("Le dernier jour de Février 2000 est: %d", $lastday);
$lastday = mktime (0,0,0,4,-31,2000);
echo strftime ("Le dernier jour de Février 2000 est: %d", $lastday);
?>
```

Voir aussi [date\(\)](#) et [time\(\)](#).

10.12.11 strftime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strftime](#) (string *format*, int *timestamp*)
[PHP 3, PHP 4]

[strftime\(\)](#) retourne la date sous la forme d'une chaîne formatée conformément au format *format*, en utilisant le timestamp *timestamp* donné. Si le *timestamp* est omis, la date actuelle est utilisée. Les mois et jours de la semaine, et toutes les chaînes dépendantes de la langue sont fixées avec la commande [setlocale\(\)](#).

Les caractères suivant sont utilisés pour spécifier le format de la date :

- %a – nom abrégé du jour de la semaine (local).
- %A – nom complet du jour de la semaine (local).
- %b – nom abrégé du mois (local).
- %B – nom complet du mois (local).
- %c – représentation préférée pour les dates et heures, en local.
- %C – Numéro de siècle (l'année, divisée par 100 et arrondie entre 00 et 99)
- %d – jour du mois en numérique (intervalle 01 à 31)
- %D – same as %m/%d/%y
- %e – numéro du jour du mois. Les chiffres sont précédés d'un espace (de ' 1' à '31')
- %h – identique à %b
- %H – heure de la journée en numérique, et sur 24–heures (intervalle 00 à 23)
- %I – heure de la journée en numérique, et sur 12– heures (intervalle 01 à 12)
- %j – jour de l'année, en numérique (intervalle 001 à 366)
- %m – mois en numérique (intervalle 1 à 12)
- %M – minute en numérique
- %n – newline character
- %p – soit 'am' ou 'pm' en fonction de l'heure absolue, ou en fonction des valeurs enregistrées en local.
- %r – l'heure au format a.m. et p.m.
- %R – l'heure au format 24h
- %S – secondes en numérique
- %t – tabulation
- %T – l'heure actuelle (égal à %H:%M:%S)

- %u – le numéro de jour dans la semaine, de 1 à 7. (1 représente Lundi)
- %U – numéro de semaine dans l'année, en considérant le premier dimanche de l'année comme le premier jour de la première semaine.
- %V – le numéro de semaine comme défini dans l'ISO 8601:1988, sous forme décimale, de 01 à 53. La semaine 1 est la première semaine qui a plus de 4 jours dans l'année courante, et dont Lundi est le premier jour.
- %W – numéro de semaine dans l'année, en considérant le premier lundi de l'année comme le premier jour de la première semaine
- %w – jour de la semaine, numérique, avec Dimanche = 0
- %x – format préféré de représentation de la date sans l'heure
- %X – format préféré de représentation de l'heure sans la date
- %y – l'année, numérique, sur deux chiffres (de 00 à 99)
- %Y – l'année, numérique, sur quatre chiffres
- %Z – fuseau horaire, ou nom ou abréviation
- %% – un caractère '%' littéral

Note : Tous les caractères suivants ne sont pas toujours supportés par toutes les librairies C. Dans ce cas, ils ne seront pas supportés par PHP non plus.

Exemple strftime()

```
setlocale ("LC_TIME", "C");
print(strftime("%A en Finlandais est "));
setlocale ("LC_TIME", "fi");
print(strftime("%A, en Français "));
setlocale ("LC_TIME", "fr");
print(strftime("%A est en Allemand "));
setlocale ("LC_TIME", "de");
print(strftime("%A.\n"));
```

Cet exemple ne fonctionnera que si vous avez les configurations respectives installées sur votre système. Voir aussi [setlocale\(\)](#) et [mktime\(\)](#) et le [groupe de spécifications de @xref{function.strftime...strftime\(\)}](#).

10.12.12 time

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [time](#)(void)

[PHP 3, PHP 4]

[time\(\)](#) retourne la heure courante, mesurée en secondes depuis le début de l'époque UNIX, (1er janvier 1970 00:00:00 GMT).

Voir aussi [date\(\)](#).

10.12.13 strtotime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strtotime](#)(string *time*, int *now*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b2]

[strtotime\(\)](#) essaye de lire une date au format anglais dans la chaîne *time*, et de la transformer en timestamp UNIX.

Exemple avec *strtotime()*

```
<?php
// l'exemple n'est pas traduit, car cela ne fonctionne qu'en anglais
echo strtotime ("now") . "\n";
echo strtotime ("10 September 2000") . "\n";
echo strtotime ("+1 day") . "\n";
echo strtotime ("+1 week") . "\n";
echo strtotime ("+1 week 2 days 4 hours 2 seconds")."\n";
?>
```

10.13 DBA

[\[Notes en ligne\]](#)

Ces fonctions sont l'interface avec les bases de type Berkeley.

C'est une couche générale pour plusieurs bases de données sur fichiers. En tant que tel, les fonctionnalités sont limitées à une partie des fonctionnalités des bases de données modernes, comme Sleepycat Software's DB2. (A ne pas confondre avec IBM's DB2 software, qui fonctionne avec [10.72 ODBC unifié](#).).

Le comportement de certaines fonctions dépend de la base de données utilisée. Par exemple [dba_optimize\(\)](#) et [dba_sync\(\)](#) n'auront pas le même effet d'une base à l'autre.

Pour ajouter le support d'une base dba, il suffit d'ajouter l'option de configuration `--with` adéquate. Les bases suivantes sont supportées :

- Dbm est la plus ancienne des base de données de type Berkeley. Il vaut mieux l'éviter si possible. Les fonctions de compatibilités codées dans DB2 et gdbm ne sont pas supportées, car elles ne sont compatibles qu'au niveau du code source, et ne peuvent pas gérer le format dbm originel.
(`--with-dbm`)
- ndbm est un nouveau type de dbm plus flexible. Il a cependant la majorité des limitations du genre.
(`--with-ndbm`)
- gdbm est la base dbm GNU. (`--with-gdbm`)
- db2 est DB2 de Sleepycat Software. Elle se décrit comme un "ensemble d'outils qui fournissent une base de données performante, tant pour les applications indépendantes que pour le client/serveur".
(`--with-db2`)
- DB3 est le [DB3 de Sleepycat Software](#). (`--with-db3`)
- cdb est "un package rapide, robuste, léger, pour créer et lire des bases de données constantes". C'est l'auteur de qmail qui l'a écrit, et elle est disponible ici. Puisque c'est une base constante, elle ne supporte que la lecture. (`--with-cdb`)

Exemple DBA

```
<?php
$id = dba_open("/tmp/test.db", "n", "db2");
if(!$id) {
echo "dba_open a échoué\n";
exit;
}
dba_replace("key", "Ceci est un exemple!", $id);
if(dba_exists("key", $id)) {
echo dba_fetch("key", $id);
}
```

```

dba_delete("key", $id);
}
dba_close($id);
?>

```

DBA gère les données binaires, et n'a pas de limites arbitraires. Elle hérite de toutes les limites de la base sous-jacentes.

Toutes les bases de données sur fichiers doivent fournir un moyen de changer le mode d'accès au fichier d'une base, et si possible, de toutes les bases. Le mode d'accès est généralement passé en 4ème argument à [dba_open\(\)](#) ou [dba_popen\(\)](#).

Vous pouvez accéder à toutes les entrées d'une base d'une manière linéaire, avec les fonctions [dba_firstkey\(\)](#) et [dba_nextkey\(\)](#). Vous ne devez pas modifier une base lorsque vous la traversez ainsi.

Passer en revue une base

```

<?php
# ...ouverture de la base...
$key = dba_firstkey($id);
while($key != FALSE) {
if(...) { # conserver la clé pour faire d'autres opérations plus tard
    $handle_later[] = $key;
}
    $key = dba_nextkey($id);
}
for($i = 0; $i < count($handle_later); $i++)
    dba_delete($handle_later[$i], $id);
?>

```

10.13.1 dba_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [dba_close](#)(int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_close\(\)](#) ferme le lien établi avec la base et libère toutes les ressources de *handle*.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_close\(\)](#) ne retourne aucune valeur.

Voir aussi: [dba_open\(\)](#) et [dba_popen\(\)](#).

10.13.2 dba_delete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_delete](#)(string *key*, int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_delete\(\)](#) efface l'entrée spécifiée par la clé *key*, dans la base identifiée par *handle*.

key est la clé de l'entrée à effacer.

handle est un identifiant de lien, retourné par [dba_open\(\)](#).

[dba_delete\(\)](#) retourne TRUE ou FALSE, si l'entrée est effacée, ou pas effacée, respectivement.

Voir aussi: [dba_exists\(\)](#), [dba_fetch\(\)](#), [dba_insert\(\)](#) et [dba_replace\(\)](#).

10.13.3 dba_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_exists](#) (string *key*, int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_exists\(\)](#) vérifie si la clé *key* existe dans la base identifiée par *handle*.

key est la clé qui doit être vérifiée.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_exists\(\)](#) retourne TRUE ou FALSE, si la clé est trouvée, ou pas, respectivement.

Voir aussi : [dba_fetch\(\)](#), [dba_delete\(\)](#), [dba_insert\(\)](#) et [dba_replace\(\)](#).

10.13.4 dba_fetch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_fetch](#) (string *key*, int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_fetch\(\)](#) lit les données spécifiée par la clé *key* dans la base identifiée par *handle*.

key est la clé dont on veut lire les données.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_fetch\(\)](#) retourne la chaîne associée ou FALSE, si la paire clé/valeur n'a pas été trouvée.

Voir aussi : [dba_exists\(\)](#), [dba_delete\(\)](#), [dba_insert\(\)](#) et [dba_replace\(\)](#).

10.13.5 dba_firstkey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_firstkey](#) (int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_firstkey\(\)](#) retourne la première clé de la base de données spécifiée par *handle* et y place le pointeur interne de clé. Cela permettra de traverser la base.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_firstkey\(\)](#) retourne la clé, ou FALSE, suivant que la première clé existe ou pas.

Voir aussi : [dba_nextkey\(\)](#).

10.13.6 dba_insert

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_insert](#) (string *key*, string *value*, int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_insert\(\)](#) insère l'entrée décrite par la clé *key* et la valeur *value* dans la base spécifiée par *handle*. Si une entrée avec la même clé *key* existe déjà, l'insertion échouera.

key est la clé de la valeur à insérer.

value est la valeur à insérer.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_insert\(\)](#) retourne TRUE ou FALSE, suivant que l'insertion a réussi ou échoué.

Voir aussi : [dba_exists\(\)](#), [dba_delete\(\)](#), [dba_fetch\(\)](#) et [dba_replace\(\)](#).

10.13.7 dba_nextkey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dba_nextkey](#)(int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_nextkey\(\)](#) retourne la clé suivante, dans la base identifiée par *handle* et incrémente le pointeur de clé.

handle est un identifiant de base, retourné par [dba_open\(\)](#).

[dba_nextkey\(\)](#) retourne la clé, ou FALSE en cas d'échec.

Voir aussi: [dba_firstkey\(\)](#).

10.13.8 dba_popen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dba_popen](#)(string *path*, string *mode*, string *handler*, ...)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_popen\(\)](#) établit une connexion persistante à la base repérée par *path* avec le mode *mode*, en utilisant l'identifiant *handler*.

path est le chemin sur votre machine.

mode vaut "r" pour lecture seule, "w" pour lecture/écriture, "c" pour lecture/écriture, et création si la base n'existe pas, et "n" pour création, écrasement, et accès en lecture/écriture.

handler est le nom de l'identifiant qui sera utilisé pour accéder à *path*. Il est passé à [dba_popen\(\)](#).

[dba_popen\(\)](#) retourne un identifiant positif, ou FALSE, suivant que la base a été ouverte, ou que l'accès a échoué.

Voir aussi : [dba_open\(\)](#) et [dba_close\(\)](#).

10.13.9 dba_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dba_open](#)(string *path*, string *mode*, string *handler*, ...)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_open\(\)](#) établit une connexion à la base repérée par *path* avec le mode *mode* et l'identifiant *handler*.

path est le chemin sur votre machine.

mode vaut "r" pour lecture seule, "w" pour lecture/écriture, "c" pour lecture/écriture, et création si la base n'existe pas, et "n" pour création, écrasement, et accès en lecture/écriture.

handler est le nom de l'identifiant qui sera utilisé pour accéder à *path*. Il est passé à [dba_popen\(\)](#).

Voir aussi : [dba_popen\(\)](#) et [dba_close\(\)](#).

10.13.10 dba_optimize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_optimize](#)(int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_optimize\(\)](#) optimise la base de données identifiée par *handle*.

handle est un identifiant de base retourné par [dba_open\(\)](#).

[dba_optimize\(\)](#) retourne TRUE ou FALSE, suivant que l'optimisation a réussi ou échoué.

Voir aussi : [dba_sync\(\)](#).

10.13.11 dba_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_replace](#)(string *key*, string *value*, int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_replace\(\)](#) remplace ou insère une entrée, pour la clé *key* et avec la valeur *value* dans la base identifiée par *handle*.

key est la clé qui va être insérée.

value est la valeur qui va être insérée.

handle est un identifiant de base retourné par [dba_open\(\)](#).

[dba_replace\(\)](#) retourne TRUE ou FALSE, suivant que l'opération réussit ou échoue.

Voir aussi : [dba_exists\(\)](#), [dba_delete\(\)](#), [dba_fetch\(\)](#) et [dba_insert\(\)](#).

10.13.12 dba_sync

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dba_sync](#)(int *handle*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[dba_sync\(\)](#) synchronise la base de données spécifiée par *handle*. Si accepté, cela va probablement lancer une opération de réécriture physique du fichier.

handle est un identifiant de base retourné par [dba_open\(\)](#).

[dba_sync\(\)](#) retourne TRUE ou FALSE, si la synchronisation réussit, ou échoue, respectivement.

Voir aussi : [dba_optimize\(\)](#).

10.14 dBase

[\[Notes en ligne\]](#)

Ces fonctions vous permettront d'accéder aux enregistrements d'une base au format dBase (.dbf).

dBase ne permet pas l'utilisation d'index, de "memo fields", ni le blocage de la base. Deux processus de serveurs web différents modifiant la même fichier dBase risque de rendre votre base de données incohérente.

A la différence des bases de données SQL, la définition des bases de données dBase, ne peut pas être changée. Une fois le fichier créé, la définition de la base est définitive. Il n'y a pas d'index qui accélèrent les recherches ou organisent vos données. Les fichiers dBase sont de simples fichiers séquentiels avec des enregistrements de longueur fixe. Les enregistrements sont ajoutés à la fin du fichier et les enregistrements supprimés sont conservés jusqu'à l'appel de [dbase_pack\(\)](#).

Nous vous recommandons de ne pas utiliser les fichiers dBase comme base de données de production.

Choisissez n'importe quel serveur SQL à la place. MySQL et Postgres sont des choix classiques avec PHP.

Le support de dBase ne se justifie ici que pour vous permettre d'importer et d'exporter des données de et vers votre base des données web, maintenant que le format du fichier est communément géré par les feuilles et

organiseurs Windows. L'import et l'export de données est l'unique chose pour laquelle l'utilisation de dBase est recommandée.

10.14.1 dbase_create

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`int dbase_create(string filename, array fields)`

[PHP 3, PHP 4]

fields est un tableau de tableaux. Chaque tableau décrit le format d'un fichier de la base. Chaque champs est constitué d'un nom, d'un caractère de type de champs, d'une longueur et d'une précision.

Les types de champs disponibles sont :

L

- Boolean (booléen). Pas de longueur ou de précision pour ces valeurs.

M

- Memo. (Note importante : les Memos ne sont pas supportés par PHP.) Elles n'ont pas de longueur ou de précision.

D

- Date (enregistrée au format 'YYYYMMDD'). Elles n'ont pas de longueur ou de précision.

N

- Number (nombre). Possède une longueur et un précision (le nombre de chiffres après la virgule).

C

- String (chaîne).

Si la base de données a été créée, un identifiant de base `dbase_identifier` est retourné, sinon, `FALSE` est retourné.

Création d'une base dBase

```
<?php
// "database" name
$dbname = "/tmp/test.dbf";
// database "definition"
$def =
array(
array("date",      "D"),
array("name",      "C",  50),
array("age",       "N",   3, 0),
array("email",     "C", 128),
array("ismember", "L")
);
// création
if (!dbase_create($dbname, $def))
print "<strong>Error!</strong>";
?>
```

[10.14.2 dbase_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbase_open](#)(string *filename*, int *flags*)

[PHP 3, PHP 4]

flags est un flag, comme pour la fonction `open()`. (Typiquement; 0 signifie lecture seule, 1 signifie écriture seule, et 2 écriture/lecture).

Retourne un identifiant de base de données, ou FALSE si la base n'a pas pu être sélectionnée.

[10.14.3 dbase_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_close](#)(int *dbase_identifieur*)

[PHP 3, PHP 4]

Ferme la base associée à *dbase_identifieur*.

[10.14.4 dbase_pack](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_pack](#)(int *dbase_identifieur*)

[PHP 3, PHP 4]

Compacte la base de données spécifiée (effacement définitif de tous les enregistrements marqués pour l'effacement, avec la fonction [dbase_delete_record\(\)](#)).

[10.14.5 dbase_add_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_add_record](#)(int *dbase_identifieur*, array *record*)

[PHP 3, PHP 4]

Ajoute les données de *record* dans la base spécifiée par *dbase_identifieur*. Si le nombre de colonnes fourni n'est pas celui du nombre de champs dans la base, l'opération échouera, et FALSE sera retourné.

[10.14.6 dbase_replace_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_replace_record](#)(int *dbase_identifieur*, array *record*, int *dbase_record_number*)

[PHP 3>= 3.0.11, PHP 4]

Remplace les données associées à l'enregistrement *record_number* par les données enregistrées dans *record*. Si le nombre de colonnes fourni n'est pas celui du nombre de champs dans la base, l'opération échouera, et FALSE sera retourné.

dbase_record_number est un entier qui peut aller de 1 jusqu'au nombre maximal d'enregistrement de la base

(retourné par [dbase_numrecords\(\)](#)).

10.14.7 dbase_delete_record

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbase_delete_record](#) (int *dbase_identifieur*, int *record*)

[PHP 3, PHP 4]

Marque l'enregistrement *record* pour l'effacement, dans la base. Pour effacer réellement l'enregistrement, il faut utiliser aussi [dbase_pack\(\)](#).

10.14.8 dbase_get_record

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [dbase_get_record](#) (int *dbase_identifieur*, int *record*)

[PHP 3, PHP 4]

Retourne les données de l'enregistrement *record* dans un tableau. Le tableau est indexé à partir de 0, et inclus un membre nommé 'deleted' (effacé), qui sera mis à 1 si l'enregistrement a été marqué pour l'effacement (voir [dbase_delete_record\(\)](#)).

Chaque champs est converti au format approprié PHP. (Les dates sont laissées au format chaîne).

10.14.9 dbase_get_record_with_names

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [dbase_get_record_with_names](#) (int *dbase_identifieur*, int *record*)

[PHP 3>= 3.0.4, PHP 4]

Retourne les données de l'enregistrement *record* dans un tableau associatif. Le tableau inclus un membre nommé 'deleted' (effacé), qui sera mis à 1 si l'enregistrement a été marqué pour l'effacement (voir [dbase_delete_record\(\)](#)).

Chaque champs est converti au format approprié PHP. (Les dates sont laissées au format chaîne).

10.14.10 dbase_numfields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbase_numfields](#) (int *dbase_identifieur*)

[PHP 3, PHP 4]

Retourne le nombre de champs (colonnes) de la base de données spécifiée. Les numéros de champs sont numérotés de 0 à `dbase_numfields($db)-1`, tandis que les numéros d'enregistrements sont numérotés de 1 à `dbase_numrecords($db)`.

Utiliser `dbase_numfields()`

```
<?php
$rec = dbase_get_record($db, $recno);
$nf  = dbase_numfields($db);
for ($i=0; $i < $nf; $i++) {
```

```
print $rec[$i]."<br>\n";
}
?>
```

10.14.11 dbase_numrecords

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbase_numrecords](#) (int *dbase_identifieur*)

[PHP 3, PHP 4]

Retourne le nombre d'enregistrements (lignes) dans la base spécifiée. Les numéros de champs sont numérotés de 0 à dbase_numfields(\$db)-1, tandis que les numéros d'enregistrements sont numérotés de 1 à dbase_numrecords(\$db).

10.15 DBM

[\[Notes en ligne\]](#)

Ces fonctions vous permettent d'écrire des lignes dans une base de donnée de type dbm. Ce type de base (supporté par Berkeley db, gdbm, et quelques librairies systèmes, ou certaines librairies du système d'exploitation) enregistre les paires clés/valeurs, (contrairement aux enregistrements par ligne, utilisé par les autres bases de données relationnelles).

dbm

```
<?php
$dbm = dbmopen("dernier", "w");
if (dbmexists($dbm, $userid)) {
    $last_seen = dbmfetch($dbm, $userid);
} else {
    dbminsert($dbm, $userid, time());
}
faire_quelquechose();
dbmreplace($dbm, $userid, time());
dbmclose($dbm);
?>
```

10.15.1 dbmopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbmopen](#) (string *filename*, string *flags*)

[PHP 3, PHP 4]

Le premier argument est le chemin absolu jusqu'au fichier dbm à ouvrir. Le deuxième argument est le mode d'ouverture du fichier, qui peut prendre les valeurs suivantes : "r", "n", "c" ou "w" qui représentent respectivement lecture seule, nouveau (ce qui implique lecture/écriture, et qui, probablement, va écraser une base existante), création(ce qui implique lecture/écriture, et qui, probablement, va écraser une base existante), et lecture/écriture.

Retourne un identifiant, qui sera passé à toutes les autres fonctions dbm, en cas de succès, ou FALSE en cas d'échec.

Si ndbm est utilisé, ndbm va créer les fichiers filename.dir et filename.pag. gdbm n'utilise qu'un fichier, tout comme les librairie internes, et Berkeley db crée le fichier filename.db. Notez que PHP dispose de son propre système de verrouillage des fichiers, qui s'additionne à celui éventuellement fait par les librairies. PHP n'efface jamais les fichiers .lck qu'il crée. Il les utilise comme inode fixe, sur lequel faire le verrouillage. Pour plus d'informations sur les fichiers dbm, reportez vous à vos pages de manuel Unix (man) , ou bien chargez gdbm : ``ftp://prep.ai.mit.edu/pub/gnu'`.

10.15.2 dbmclose

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmclose](#) (int *dbm_identifieur*)

[PHP 3, PHP 4]

Déverrouille et ferme la base de données dbm_identifieur.

10.15.3 dbmexists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmexists](#) (int *dbm_identifieur*, string *key*)

[PHP 3, PHP 4]

Retourne TRUE si il y a une valeur associée à la clé *key*.

10.15.4 dbmfetch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dbmfetch](#) (int *dbm_identifieur*, string *key*)

[PHP 3, PHP 4]

Retourne la valeur associée à la clé *key*.

10.15.5 dbminsert

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dbminsert](#) (int *dbm_identifieur*, string *key*, string *value*)

[PHP 3, PHP 4]

Ajoute la valeur *value* dans la base de données, avec la clé *key*.

Retourne -1 si la base a été ouverte en mode lecture seule, 0 si l'insertion a été réussie, et 1 si la clé *key* existe déjà. (Pour remplacer la valeur, utilisez [dbmreplace\(\)](#).)

10.15.6 dbmreplace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmreplace](#) (int *dbm_identifieur*, string *key*, string *value*)

[PHP 3, PHP 4]

Remplace une valeur courante par la valeur *value* pour la clé *key*, dans une base dbm.
 Cette fonction va créer la clé, si elle n'existe pas dans la base.

[10.15.7 dbmdelete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [dbmdelete](#) (int *dbm_identifieur*, string *key*)

[PHP 3, PHP 4]

Efface la valeur de la clé *key*, dans la base dbm.
 Retourne FALSE si la clé n'existe pas dans cette base.

[10.15.8 dbmfirstkey](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dbmfirstkey](#) (int *dbm_identifieur*)

[PHP 3, PHP 4]

Retourne la première clé de la base de données. Notez bien que les clés ne sont pas dans un ordre défini, étant donné que la table est construite comme une table de hash.

[10.15.9 dbmnextkey](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dbmnextkey](#) (int *dbm_identifieur*, string *key*)

[PHP 3, PHP 4]

[dbmnextkey\(\)](#) retourne la clé après la clé *key*. En appelant [dbmfirstkey\(\)](#), puis successivement [dbmnextkey\(\)](#), il est possible de passer en revue toute les paires clé/valeur de la base de données dbm. Par exemple :

Passer en revue une base de données.

```
<?php
$key = dbmfirstkey($dbm_id);
while ($key) {
    echo "$key = " . dbmfetch($dbm_id, $key) . "\n";
    $key = dbmnextkey($dbm_id, $key);
}
?>
```

[10.15.10 dblist](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dblist](#) (void)

[PHP 3, PHP 4]

10.16 Accès aux dossiers

[\[Notes en ligne\]](#)

10.16.1 chdir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chdir](#) (string *directory*)
[PHP 3, PHP 4]

[chdir\(\)](#) change le dossier courant de PHP en *directory*. [chdir\(\)](#) retourne FALSE si l'opération échoue, et TRUE sinon.

10.16.2 dir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

new [dir](#) (string *directory*)
[PHP 3, PHP 4]

Un mécanisme pseudo—objet permet la lecture d'un dossier. L'argument *directory* doit être ouvert. Deux propriétés sont disponibles une fois le dossier ouvert : le pointeur peut être utilisé avec d'autres fonctions telles que [readdir\(\)](#), [rewinddir\(\)](#) et [closedir\(\)](#). Le chemin du dossier est le chemin fourni lors de la construction de l'objet. Trois méthodes permettent de lire, remettre à zéro et fermer le dossier.

Exemple avec dir()

```
<?php
$d = dir("/etc");
echo "Pointeur: ".$d->handle."<br>\n";
echo "Chemin: ".$d->path."<br>\n";
while($entry=$d->read()) {
echo $entry."<br>\n";
}
$d->close();
?>
```

10.16.3 closedir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [closedir](#) (int *dir_handle*)
[PHP 3, PHP 4]

[closedir\(\)](#) ferme le pointeur de dossier *dir_handle*. Le dossier devait avoir été ouvert avec [opendir\(\)](#).

10.16.4 getcwd

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getcwd](#)

[PHP 4 >= 4.0b4]

[getcwd\(\)](#) retourne le nom du dossier courant.

[10.16.5 opendir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [opendir](#)(string *path*)

[PHP 3, PHP 4]

[opendir\(\)](#) retourne un pointeur sur un dossier pour être utilisé avec les fonctions [closedir\(\)](#), [readdir\(\)](#) et [rewinddir\(\)](#).

[10.16.6 readdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [readdir](#)(int *dir_handle*)

[PHP 3, PHP 4]

[readdir\(\)](#) retourne le nom du fichier suivant dans le dossier identifié par *dir_handle*. Les noms sont retournés dans n'importe quel ordre.

Liste tous les fichiers du dossier courant

```
<?php
$handle=opendir('.');
echo "Pointeur de dossier: $handle\n";
echo "Fichiers:\n";
while ($file = readdir($handle)) {
    echo "$file\n";
}
closedir($handle);
?>
```

Notez que [readdir\(\)](#) retournera aussi les dossiers "." et "..". Si vous ne les voulez pas, supprimez les simplement :

Liste tous les fichiers du dossier courant, sauf . et ..

```
<?php
$handle=opendir('.');
while ($file = readdir($handle)) {
    if ($file != "." 38;$file != "..") {
        echo "$file\n";
    }
}
closedir($handle);
?>
```

[10.16.7 rewinddir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [rewinddir](#) (int *dir_handle*)

[PHP 3, PHP 4]

[rewinddir\(\)](#) retourne à la première entrée du dossier : le prochain fichier lu sera le premier.

[10.17 DOM XML](#)

[\[Notes en ligne\]](#)

Ces fonctions ne sont disponibles que si PHP a été configuré avec l'option `--with-dom=[DIR]`, et utilise la librairie GNOME xml library. Vous aurez aussi besoin de la librairie libxml-2.0.0 (la version beta ne fonctionne pas). Ces fonctions ont été ajoutées dans PHP 4.

Ce module définit les constantes suivantes :

Constante	Valeur	Description
XML_ELEMENT_NODE	1	@tab
XML_ATTRIBUTE_NODE	2	@tab
XML_TEXT_NODE	3	@tab
XML_CDATA_SECTION_NODE	4	@tab
XML_ENTITY_REF_NODE	5	@tab
XML_ENTITY_NODE	6	@tab
XML_PI_NODE	7	@tab
XML_COMMENT_NODE	8	@tab
XML_DOCUMENT_NODE	9	@tab
XML_DOCUMENT_TYPE_NODE	10	@tab
XML_DOCUMENT_FRAG_NODE	11	@tab
XML_NOTATION_NODE	12	@tab
XML_GLOBAL_NAMESPACE	1	@tab
XML_LOCAL_NAMESPACE	2	@tab

Ce module définit un ensemble de classes. Les fonctions DOM XML retourne un arbre à partir d'un document XML, dont chaque noeud est un objet issu de ces classes.

[10.17.1 xmldoc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [xmldoc](#) (string *str*)

[PHP 4 >= 4.0b4]

[xmldoc\(\)](#) analyse le document XML *str* et retourne un objet de classe "Dom document", avec les propriétés de "doc" (ressources), "version" (string) et "type" (long).

[10.17.2 xmldocfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [xmldocfile](#) (string *filename*)
[PHP 4 >= 4.0b4]

[xmldocfile\(\)](#) analyse le fichier XML *filename* et retourne un objet "Dom document", avec les propriétés de "doc" (ressources) et "version" (string).

[10.17.3 xmltree](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [xmltree](#) (string *str*)
[PHP 4 >= 4.0b4]

[xmltree\(\)](#) analyse le document XML *str* et retourne un arbre d'objets PHP qui représente le document analysé.

[10.18 Gestion des erreurs](#)

[\[Notes en ligne\]](#)

Ces fonctions permettent de gérer les erreurs, et de les enregistrer. Vous pouvez définir les règles de traitement des erreurs et choisir la manière de les enregistrer : vous pouvez adapter le rapport d'erreur à vos besoins.

Avec les fonctions d'enregistrements, vous pouvez envoyer directement les rapport à d'autres machines (ou même les envoyer par email à un pager), à l'historique système, ou encore sélectionner les erreurs les plus importantes et ne pas enregistrer les autres.

La fonction de niveau d'erreur vous permet de personnaliser le niveau et le type d'erreur noté : depuis les inoffensives alertes jusqu'au erreurs personnalisées retournées par les fonctions.

[10.18.1 error_log](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [error_log](#) (string *message*, int *message_type*, string *destination* , string *extra_headers*)
[PHP 3, PHP 4]

[error_log\(\)](#) envoie un message d'erreur à l'historique du serveur web, à un port *TCP* ou un fichier. *message* est le message d'erreur qui doit être enregistré. *message_type* indique où le message doit être envoyé :

0	<i>message</i> est envoyé à l'historique PHP, qui est basé sur l'historique système ou un fichier, en fonction de la configuration de 7.1.1.9 ini.error-log . @tab
1	<i>message</i> est envoyé par email à l'adresse <i>destination</i> . C'est le seul type qui utilise le quatrième paramètre

	<i>extra_headers</i> . Ce message utilise la même fonction interne que mail() . @tab
2	<i>message</i> est envoyé par la connexion de debuggage PHP. Cette option n'est disponible que si l'option 4.3.4.13 install.configure.enable-debugger a été désactivé. Dans ce cas, le paramètre <i>destination</i> spécifie l'hôte ou l'adresse IP, et optionnellement le numéro de port, de la socket qui recevra les informations de débogage. @tab
3	<i>message</i> est ajouté au fichier <i>destination</i> . @tab

Exemples avec `error_log()`

```
<?php
// Envoi une notification par l'historique du serveur, si la connexion à la base
// de données est impossible.
if (!Ora_Logon ($username, $password)) {
error_log ("Base Oracle indisponible!", 0);
}
// Indiquer à l'administrateur, par email, qu'il n'y a plus de FOO
if (!($foo = allocate_new_foo())) {
error_log ("Aya!, Il ne reste plus de FOO disponibles!", 1,
          "opérateur@mondomaine.com");
}
// D'autres manières d'appeler error_log():
error_log ("Grosse bourde!", 2, "127.0.0.1:7000");
error_log ("Grosse bourde!", 2, "loghost");
error_log ("Grosse bourde!", 3, "/var/tmp/my-errors.log");
?>
```

10.18.2 error_reporting

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [error_reporting](#) (int *level*)

[PHP 3, PHP 4]

[error_reporting\(\)](#) fixe le niveau de rapport d'erreur PHP et retourne l'ancienne valeur. Le niveau d'erreur peut être un champs de bits, ou une constante. L'utilisation des constantes est vivement recommandé, pour assurer une compatibilité maximale avec les futures versions. Au fur et à mesure que de nouveaux niveaux d'erreurs sont créés, l'intervalle de validité des niveaux évolue, et les anciennes valeurs n'ont plus les mêmes significations.

Exemple de modification de niveau d'erreur

```
error_reporting (55); // En PHP 3, équivalent à E_ALL ^ E_NOTICE
/* ...en PHP 4, '55' signifie (E_ERROR | E_WARNING | E_PARSE |
E_CORE_ERROR | E_CORE_WARNING) */
error_reporting (2039); // PHP 4 équivalent à E_ALL ^ E_NOTICE
error_reporting (E_ALL ^ E_NOTICE); // La même signification en PHP 3 et 4
```

Suivez les liens de chaque valeur interne pour connaître leur signification :

constante	valeur
-----------	--------

1	C.3.3 E_ERROR @tab
2	C.3.2 E_WARNING @tab
4	C.3.4 E_PARSE @tab
8	C.3.1 E_NOTICE @tab
16	C.3.5 E_CORE_ERROR @tab
32	C.3.6 E_CORE_WARNING @tab
64	C.3.7 E_COMPILE_ERROR @tab
128	C.3.8 E_COMPILE_WARNING @tab
256	C.3.9 E_USER_ERROR @tab
512	C.3.10 E_USER_WARNING @tab
1024	C.3.9 E_USER_ERROR @tab

Exemples avec error_reporting()

```

error_reporting(0);
/* Empêche tout affichage d'erreur */
error_reporting(7); // Ancienne syntaxe PHP 2/3
error_reporting(E_ERROR | E_WARNING | E_PARSE); // Nouvelle syntaxe PHP 3/4
/* Utilisation appropriée pour les erreurs courantes d'exécution */
error_reporting(15); // Ancienne syntaxe, PHP 2/3
error_reporting(E_ERROR | E_WARNING | E_PARSE | E_NOTICE); // Nouvelle syntaxe PHP 3/4
/* Utilisation appropriée pour les erreurs courantes de développement
   (variables non initialisées..) */
error_reporting(63); // Ancienne syntaxe, PHP 2/3
error_reporting(E_ALL); // Nouvelle syntaxe PHP 3/4
/* rapporte toutes les erreurs PHP*/

```

10.18.3 restore_error_handler

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [restore_error_handler](#)(void)

[PHP 4 >= 4.0.1]

Utilisée après avoir modifié la fonction de gestion des erreurs, grâce à [set_error_handler\(\)](#), [restore_error_handler\(\)](#) permet de réutiliser l'ancienne version de gestion des erreurs (qui peut être la fonction PHP par défaut, ou une autre fonction utilisateur).

Voir aussi [error_reporting\(\)](#), [set_error_handler\(\)](#), [trigger_error\(\)](#) et [user_error\(\)](#).

10.18.4 set_error_handler

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [set_error_handler](#)(string *error_handler*)

[PHP 4 >= 4.0.1]

[set_error_handler\(\)](#) choisit la fonction utilisateur *error_handler* pour gérer les erreurs dans un script.

Retourne un pointeur sur l'ancienne fonction de gestion des erreurs (si il y en avait une), ou FALSE, en cas d'erreur. [set_error_handler\(\)](#) sert à définir votre propre gestionnaire d'erreur, qui prendra en charge leur traitement durant l'exécution d'un script. Cela peut être utile lorsque vous devez repérer des erreurs critiques lors d'un nettoyage de bases, ou bien si vous souhaitez générer une erreur dans certaines conditions (avec [trigger_error\(\)](#)).

La fonction utilisateur doit accepter deux arguments : le code de l'erreur, et une chaîne décrivant l'erreur.

L'exemple ci dessous montre le traitement d'exceptions en déclenchant des erreurs, et en les gérant avec une fonction utilisateur :

Traitement des erreurs avec set_error_handler() et trigger_error()

```
<?php
// redéfinit les constantes utilisateurs - PHP 4 seulement
define (FATAL,E_USER_ERROR);
define (ERROR,E_USER_WARNING);
define (WARNING,E_USER_NOTICE);
// Fixe le niveau de rapport d'erreur pour ce script
error_reporting (FATAL + ERROR + WARNING);
// Fonction de traitement des erreurs
function myErrorHandler ($errno, $errstr) {
switch ($errno) {
case FATAL:
echo "<B>FATAL</B> [$errno] $errstr<br>\n";
echo " Erreur fatale à la ligne ".__LINE__." du fichier ".__FILE__;
echo ", PHP ".__PHP_VERSION__ ( ".__PHP_OS__ )<br>\n";
echo "Aborting...<br>\n";
exit -1;
break;
case ERROR:
echo "<B>ERREUR</B> [$errno] $errstr<br>\n";
break;
case WARNING:
echo "<B>ALERTE</B> [$errno] $errstr<br>\n";
break;
default:
echo "Erreur inconnue de type : [$errno] $errstr<br>\n";
break;
}
}
// fonction qui teste la gestion d'erreur
function scale_by_log ($vect, $scale) {
if ( !is_numeric($scale) || $scale <= 0 )
trigger_error("log(x) pour x <= 0 est indéfini, vous avez passé: scale = $scale",
FATAL);
if (!is_array($vect)) {
trigger_error("Vecteur d'entrée incorrect : un tableau de valeurs est attendu : ", ERROR);
return null;
}
for ($i=0; $i<count($vect); $i++) {
if (!is_numeric($vect[$i]))
trigger_error("La valeur à la position $i n'est pas un nombre. On utilise 0 (zéro) à la place",
WARNING);
$temp[$i] = log($scale) * $vect[$i];
}
return $temp;
}
// Ancienne fonction de traitement des erreurs
```

```
$old_error_handler = set_error_handler("myErrorHandler");
// Génération de quelques erreurs : définition d'un tableau avec des éléments non numériques
echo "vector a\n";
$a = array(2,3,"foo",5.5,43.3,21.11);
print_r($a);
// définition d'un deuxième tableau à problème
echo "----\nvector b - a alerte (b = log(PI) * a)\n";
$b = scale_by_log($a, M_PI);
print_r($b);
// Ceci est un problème, on passe une chaîne à la place d'un tableau
echo "----\nvector c - une erreur\n";
$c = scale_by_log("not array",2.3);
var_dump($c);
// Ceci est critique : le tableau contient des valeurs négatives
echo "----\nvector d - fatal error\n";
$d = scale_by_log($a, -2.5);
?>
```

L'exécution du script devrait donner ceci :

```
vector a
Array
(
    [0] => 2
    [1] => 3
    [2] => foo
    [3] => 5.5
    [4] => 43.3
    [5] => 21.11
)
----
vector b - une alerte (b = log(PI) * a)
<B>WARNING</B> [1024] La valeur à la position 2 n'est pas un nombre. On utilise 0 (zéro) à la place
Array
(
    [0] => 2.2894597716988
    [1] => 3.4341896575482
    [2] => 0
    [3] => 6.2960143721717
    [4] => 49.566804057279
    [5] => 24.165247890281
)
----
vector c - an error
<B>ERROR</B> [512] Vecteur d'entrée incorrect : un tableau de valeur est attendu<br>
NULL
----
vector d - fatal error
<B>FATAL</B> [256] log(x) de x <= 0 est indéfini : scale = -2.5<br>
Erreur fatale à la ligne 16 du fichier trigger_error.php, PHP 4.0.1pl2 (Linux)<br>
Annulation du script....<br>
```

Il faut se rappeler que la fonction standard de traitement des erreurs de PHP est alors complètement ignorée. [error_reporting\(\)](#) n'aura plus d'effet, et votre fonction de gestion des erreurs sera toujours appelée. Vous pourrez toujours lire la valeur de l'erreur courante de [error_reporting\(\)](#) et faire réagir la fonction de gestion des erreurs en fonction. Cette remarque est notamment valable si la commande a été préfixée par [9.7.5 Opérateur de contrôle d'erreur](#) (0 sera retourné).

Notez aussi qu'il est alors confié à cette fonction de terminer le script ([die\(\)](#)) si nécessaire. Si la fonction de gestion des erreurs se termine normalement, l'exécution du script se poursuivra avec l'exécution de la

prochaine commande.

Voir aussi [error_reporting\(\)](#), [restore_error_handler\(\)](#), [trigger_error\(\)](#), et [user_error\(\)](#).

10.18.5 trigger_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [trigger_error](#) (string *error_msg*, int *error_type*)

[PHP 4 >= 4.0.1]

[trigger_error\(\)](#) est utilisé pour déclencher une erreur utilisateur. Elle peut aussi être utilisée en conjonction avec un gestionnaire d'erreur interne, ou un gestionnaire d'erreurs utilisateur qui a été choisi comme gestionnaire d'erreur avec [set_error_handler\(\)](#).

[trigger_error\(\)](#) est pratique lorsque vous devez générer une réponse particulière lors de l'exécution. Par exemple

```
<?php
if (assert ($divisor == 0))
trigger_error ("Impossible de diviser par zéro", E_USER_ERROR);
?>
```

Note : Voir [set_error_handler\(\)](#) pour illustration.

Voir aussi [error_reporting\(\)](#), [set_error_handler\(\)](#), [restore_error_handler\(\)](#), [user_error\(\)](#)

10.18.6 user_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [user_error](#) (string *error_msg*, int *error_type*)

[PHP 4 >= 4.0RC2]

[user_error\(\)](#) est un alias de la fonction [trigger_error\(\)](#).

Voir aussi [error_reporting\(\)](#), [set_error_handler\(\)](#), [restore_error_handler\(\)](#) et [trigger_error\(\)](#).

10.19 Exécution de programmes externes

[\[Notes en ligne\]](#)

10.19.1 escapeshellarg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [escapeshellarg](#) (string *arg*)

[PHP 4 >= 4.0.3]

[escapeshellarg\(\)](#) ajoute des guillemets simples autour des chaînes de caractères, et ajoute des guillemets puis échappe les guillemets simples de la chaîne. Cela permet de faire passer directement une chaîne comme argument shell, tout en assurant un maximum de sécurité. Cette fonction doit être utilisée pour traiter individuellement chacun des arguments à passer au shell. Les fonctions shell sont [exec\(\)](#), [system\(\)](#) est les opérateur de [9.7.6 Opérateur d'exécutions](#). Une utilisation typique est :


```
system("ls ".EscapeShellArg($dir))
```

Voir aussi [exec\(\)](#), [popen\(\)](#), [system\(\)](#), et les opérateurs de [9.7.6 Opérateur d'exécutions](#).

10.19.2 escapeshellcmd

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [escapeshellcmd\(\)](#) (string *command*)

[PHP 3, PHP 4]

[escapeshellcmd\(\)](#) échappe tous les caractères de la chaîne *command* qui pourraient avoir une signification spéciale dans une commande shell. Cette fonction permet de s'assurer que la commande sera correctement passée à l'exécuteur de commande shell [exec\(\)](#) et [system\(\)](#), ou encore à [9.7.6 Opérateur d'exécutions](#). Généralement, cette fonction est utilisée comme ceci :

```
system(EscapeShellCmd($cmd))
```

Voir aussi [exec\(\)](#), [popen\(\)](#), [system\(\)](#), et les [9.7.6 Opérateur d'exécutions](#).

10.19.3 exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [exec\(\)](#) (string *command*, string *array* , int *return_var*)

[PHP 3, PHP 4]

[exec\(\)](#) exécute la commande *command*, mais ne renvoie rien comme retour, hormis la dernière ligne du résultat de la commande. Pour exécuter une commande et obtenir le résultat sans aucun traitement, il faut utiliser la fonction [passthru\(\)](#).

Si l'argument *array* est présent, alors ce tableau sera rempli par les lignes retournées par la commande. Il faut noter que si ce tableau contient des éléments, [exec\(\)](#) ajoutera les nouvelles lignes à la fin du tableau. Si vous ne voulez pas que les nouveaux éléments soient concaténés, utilisez la fonction [unset\(\)](#) avec ce tableau avant de le passer à [exec\(\)](#).

Si l'argument *return_var* est présent en plus du tableau *array*, alors de statut de retour d'exécution sera inscrit dans cette variable.

Notez que si vous allez fournir des commandes qui proviennent d'un utilisateur, il est avisé d'utiliser la fonction [escapeshellcmd\(\)](#) pour s'assurer que l'utilisateur n'essaie pas de profiter des caractères spéciaux pour tromper le système.

Voir aussi [system\(\)](#), [passthru\(\)](#), [popen\(\)](#), [escapeshellcmd\(\)](#), et les [9.7.6 Opérateur d'exécutions](#).

10.19.4 passthru

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [passthru\(\)](#) (string *command*, int *return_var*)

[PHP 3, PHP 4]

La fonction [passthru\(\)](#) est similaire à la fonction [exec\(\)](#) car les deux exécutent la commande *command*. Si l'argument *return_var* est présent, le code de statut de réponse UNIX y sera placé. Cette fonction doit être

utilisée de préférence aux commandes [exec\(\)](#) ou [system\(\)](#) lorsque le résultat attendu est de type binaire, et doit être passé tel quel à un navigateur. Une utilisation classique de cette fonction est l'exécution de l'utilitaire pbmplus qui peut retourner une image. En fixant le résultat du contenu (content-type) à "image/gif" puis en appelant pbmplus pour obtenir une image gif, vous pouvez créer des scripts PHP qui retournent des images. Voir aussi [exec\(\)](#), [system\(\)](#), [popen\(\)](#), [escapeshellcmd\(\)](#), et les [9.7.6 Opérateur d'exécutions](#).

10.19.5 system

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [system](#) (string *command*, int *return_var*)
[PHP 3, PHP 4]

[system\(\)](#) est la version PHP de la fonction C qui exécute la commande *command* et retourne le résultat. Si une variable est fournie comme second argument, alors le code de statut de la commande y sera affecté. Notez que si vous allez fournir des commandes qui proviennent d'un utilisateur, il est avisé d'utiliser la fonction [escapeshellcmd\(\)](#) pour s'assurer que l'utilisateur n'essaie pas de profiter des caractères spéciaux pour tromper le système.

[system\(\)](#) essaie automatiquement de vider les tampons du serveur web après chaque ligne de résultat PHP, lorsque ce dernier fonctionne comme un module.

Retourne la dernière ligne du retour, en cas de succès, et FALSE en cas d'échec.

Si vous devez exécuter une commande et récupérer tout le résultat dans aucune intervention, utilisez la fonction [passthru\(\)](#).

Voir aussi [exec\(\)](#), [passthru\(\)](#), [popen\(\)](#), [escapeshellcmd\(\)](#) et les [9.7.6 Opérateur d'exécutions](#).

10.20 Forms Data Format

[\[Notes en ligne\]](#)

Forms Data Format (FDF) est un format de formulaire pour les documents PDF. Vous pouvez lire la documentation (en anglais) à <http://partners.adobe.com/asn/developer/acroSDK/forms.html> pour plus de détails sur les tenants et les aboutissants.

Note : Si vous rencontrez des problèmes de configuration de PHP avec le support fdf, vérifiez bien que le fichier d'entêtes FdfTk.h et la librairie libFdfTk.so sont bien situés. Elle devrait être dans les dossiers fdf-tk-dir/include et fdf-tk-dir/lib. Cela ne sera pas le cas si vous avez simplement décompressé la distribution FdfTk.

L'esprit de FDF est similaire à celui des formulaires HTML. Les différences résident dans les moyens de transmission des données au serveur, lorsque le bouton "submit" (soumettre) est pressé (ce qui est du ressort de Form Data Format) et le format de formulaire lui-même (qui est plutôt du ressort de Portable Document Format, PDF). Gérer des données FDF est un des objectifs des fonctions FDF. Mais il y en a d'autres. Vous pouvez aussi prendre un formulaire PDF, et pré-remplir les champs, sans modifier le formulaire lui-même. Dans ce cas, on va créer un document FDF ([fdf_create\(\)](#)), remplir les champs ([fdf_set_value\(\)](#)) et l'associer à un fichier PDF ([fdf_set_file\(\)](#)). Finalement, le tout sera envoyé au client, avec le type MIME "application/vnd.fdf". Le module "Acrobat reader" de votre navigateur va reconnaître ce type MIME, et lire le fichier PDF, puis le remplir avec FDF.

Si vous éditez un fichier FDF avec un éditeur de texte, vous trouverez un catalogue d'objet avec le nom de FDF. Cet objet peut contenir des entrées telles que Fields, F, Status etc.. Les entrées les plus couramment utilisées sont Fields, qui indique une liste de champs de contrôle, et F qui contient le nom du fichier PDF à qui appartiennent ces données. Ces entrées sont désignées dans la documentation PDF sous le nom de /F-Key ou /Status-Key. La modification de ces entrées est possible avec les fonctions [fdf_set_file\(\)](#) et [fdf_set_status\(\)](#). Les champs sont modifiables avec les fonctions [fdf_set_value\(\)](#), [fdf_set_opt\(\)](#) etc..

Les exemples suivants montre comme évaluer les données du formulaire.

Evaluer un document FDF

```
<?php
// Sauver le fichier FDF dans un fichier temporaire.
$fdffp = fopen("test.fdf", "w");
fwrite($fdffp, $HTTP_FDF_DATA, strlen($HTTP_FDF_DATA));
fclose($fdffp);
// Ouvrir le fichier temporaire, et utiliser les données.
// Le formulaire pdf contenait différents fichiers texte, avec pour nom :
// volume, date, comment, publisher, preparer, ainsi que deux boîtes
// à cocher show_publisher et show_preparer.
$fdf = fdf_open("test.fdf");
$volume = fdf_get_value($fdf, "volume");
echo "La valeur du champs volume était : '<B>$volume</B>'<BR>";
$date = fdf_get_value($fdf, "date");
echo "La valeur du champs date était '<B>$date</B>'<BR>";
$comment = fdf_get_value($fdf, "comment");
echo "La valeur du champs comment était '<B>$comment</B>'<BR>";
if(fdf_get_value($fdf, "show_publisher") == "On") {
    $publisher = fdf_get_value($fdf, "publisher");
    echo "La valeur du champs publisher était : '<B>$publisher</B>'<BR>";
} else
    echo "La valeur du champs ne doit pas être affichée.<BR>";
if(fdf_get_value($fdf, "show_preparer") == "On") {
    $preparer = fdf_get_value($fdf, "preparer");
    echo "La valeur du champs preparer était '<B>$preparer</B>'<BR>";
} else
    echo "La valeur du champs Preparer ne doit pas être affiché.<BR>";
fdf_close($fdf);
?>
```

10.20.1 fdf_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fdf_open](#) (string *filename*)

[PHP 3>= 3.0.6, PHP 4]

[fdf_open\(\)](#) ouvre un fichier avec formulaire. Le fichier doit contenir les données retournées par le formulaire PDF. Actuellement, le fichier doit être créée 'manuellement', en utilisant la fonction [fopen\(\)](#) et en y écrivant le contenu du tableau HTTP_FDF_DATA avec la fonction [fwrite\(\)](#). Un mécanisme comparable aux formulaires HTML qui créent une variable pour chaque champs entrant, n'existe pas.

Accéder aux données du formulaire

```
<?php
// Sauver le fichier FDF dans un fichier temporaire.
$fdffp = fopen("test.fdf", "w");
fwrite($fdffp, $HTTP_FDF_DATA, strlen($HTTP_FDF_DATA));
fclose($fdffp);
// Ouvrir le fichier temporaire, et utiliser les données.
$fdf = fdf_open("test.fdf");
...
fdf_close($fdf);
?>
```

Voir aussi [fdf_close\(\)](#).

10.20.2 fdf_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_close](#)(int *fdf_document*)

[PHP 3>= 3.0.6, PHP 4]

[fdf_close\(\)](#) ferme le document FDF.

Voir aussi [fdf_open\(\)](#).

10.20.3 fdf_create

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fdf_create](#)(void)

[PHP 3>= 3.0.6, PHP 4]

[fdf_create\(\)](#) crée un nouveau document FDF. Cette fonction est nécessaire pour ceux qui veulent pré remplir les champs d'un formulaire dans un fichier PDF.

Pré remplir un formulaire PDF

```
<?php
$outfdf = fdf_create();
fdf_set_value($outfdf, "volume", $volume, 0);
fdf_set_file($outfdf, "http://testfdf/resultlabel.pdf");
fdf_save($outfdf, "outtest.fdf");
fdf_close($outfdf);
Header("Content-type: application/vnd.fdf");
$fp = fopen("outtest.fdf", "r");
fpassthru($fp);
unlink("outtest.fdf");
?>
```

Voir aussi [fdf_close\(\)](#), [fdf_save\(\)](#), et [fdf_open\(\)](#).

10.20.4 fdf_save

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fdf_save](#)(string *filename*)

[PHP 3>= 3.0.6, PHP 4]

[fdf_save\(\)](#) sauve un document FDF. Le FDF Toolkit fournit un moyen d'envoyer le contenu d'un document FDF à au fichier de sortie stdout si le paramètre *filename* vaut '!. Ceci ne fonctionne pas si PHP est sous la forme d'un module Apache. Dans ce cas, il faudra écrire le résultat dans un fichier, et utiliser [fpassthru\(\)](#) pour l'afficher au client.

Voir aussi [fdf_close\(\)](#) et pour avoir un exemple [fdf_create\(\)](#).

10.20.5 fdf_get_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_get_value](#)(int *fdf_document*, string *fieldname*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_get_value\(\)](#) retourne la valeur d'un champs.
Voir aussi [fdf_set_value\(\)](#).

10.20.6 fdf_set_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_value](#)(int *fdf_document*, string *fieldname*, string *value*, int *isName*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_set_value\(\)](#) fixe la valeur d'un champs. Le dernier paramètre détermine si la valeur doit être convertie en nom PDF (*isName* = 1) ou affecter une chaîne PDF à un contrôle (*isName* = 0).
Voir aussi [fdf_get_value\(\)](#).

10.20.7 fdf_next_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_next_field_name](#)(int *fdf_document*, string *fieldname*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_next_field_name\(\)](#) retourne le nom du champs après le champs *fieldname* ou le nom du premier champs, si le second paramètre est NULL.
Voir aussi [fdf_set_value\(\)](#) et [fdf_get_value\(\)](#).

10.20.8 fdf_set_ap

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_ap](#)(int *fdf_document*, string *field_name*, int *face*, string *filename*, int *page_number*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_set_ap\(\)](#) fixe l'apparence d'un champs (i.e. la valeur de la clé /AP). Les valeurs possibles de *face* sont 1=FDFNormalAP, 2=FDFRolloverAP, 3=FDFDownAP.

10.20.9 fdf_set_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_status](#)(int *fdf_document*, string *status*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_set_status\(\)](#) fixe la valeur de la clé /STATUS.
Voir aussi [fdf_get_status\(\)](#).

10.20.10 fdf_get_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_get_status](#)(int *fdf_document*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_get_status\(\)](#) retourne la valeur de la clé /STATUS.
Voir aussi [fdf_set_status\(\)](#).

10.20.11 fdf_set_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_file](#)(int *fdf_document*, string *filename*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_set_file\(\)](#) Fixe la valeur de la clé /F. la clé /F est simplement une référence sur un formulaire PDF qui doit être pré-remplis. Dans un environnement web, c'est une URL (e.g. <http://testfdf/resultlabel.pdf>).
Voir aussi [fdf_get_file\(\)](#) et pour un exemple, [fdf_create\(\)](#).

10.20.12 fdf_get_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fdf_get_file](#)(int *fdf_document*)
[PHP 3>= 3.0.6, PHP 4]

[fdf_get_file\(\)](#) lit la valeur de la clé /F.
Voir aussi [fdf_set_file\(\)](#).

10.20.13 fdf_set_flags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_flags](#)(int *fdf_document*, string *fieldname*, int *whichFlags*, int *newFlags*)
[PHP 4 >= 4.0.2]

[fdf_set_flags\(\)](#) modifie certaines options du champs *fieldname*.
Voir aussi [fdf_set_opt\(\)](#).

10.20.14 fdf_set_opt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_opt](#)(int *fdf_document*, string *fieldname*, int *element*, string *str1*, string *str2*)
[PHP 4 >= 4.0.2]

[fdf_set_opt\(\)](#) modifie les options du champs *fieldname*.

Voir aussi [fdf_set_flags\(\)](#).

10.20.15 fdf_set_submit_form_action

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_submit_form_action](#)(int *fdf_document*, string *fieldname*, int *trigger*, string *script*, int *flags*)
[PHP 4 >= 4.0.2]

[fdf_set_submit_form_action\(\)](#) affecte un javascript au champs *fieldname*, exécuté lors de la validation d'un formulaire.

Voir aussi [fdf_set_javascript_action\(\)](#).

10.20.16 fdf_set_javascript_action

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [fdf_set_javascript_action](#)(int *fdf_document*, string *fieldname*, int *trigger*, string *script*)
[PHP 4 >= 4.0.2]

The [fdf_set_javascript_action\(\)](#) affecte un javascript au champs *fieldname*, exécuté lors de la validation d'un formulaire.

Voir aussi [fdf_set_submit_form_action\(\)](#).

10.21 FilePro

[\[Notes en ligne\]](#)

Ces fonctions permettent de lire des données enregistrées dans des bases non modifiables, sur des serveurs filePro.

filePro est une marque de Fiserv, Inc. Vous pouvez avoir plus de détails sur filePro à

<http://www.fileproplus.com/>.

10.21.1 filepro

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [filepro](#)(string *directory*)
[PHP 3, PHP 4]

[filepro\(\)](#) lit et vérifie un fichier, puis enregistre le nombre de champs et de lignes.

Aucun verrouillage n'est pratiqué : il vaut alors mieux ne pas modifier la base filePro lorsqu'elle est ouverte par PHP.

10.21.2 filepro_fieldname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filepro_fieldname](#)(int *field_number*)

[PHP 3, PHP 4]

[filepro_fieldname\(\)](#) retourne le nom du champs d'index *field_number*.

10.21.3 filepro_fieldtype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filepro_fieldtype](#) (int *field_number*)

[PHP 3, PHP 4]

Retourne le type du champs d'index *field_number*.

10.21.4 filepro_fieldwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filepro_fieldwidth](#) (int *field_number*)

[PHP 3, PHP 4]

Retourne la taille du champs d'index *field_number*.

10.21.5 filepro_retrieve

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filepro_retrieve](#) (int *row_number*, int *field_number*)

[PHP 3, PHP 4]

[filepro_retrieve\(\)](#) retourne la valeur du champs d'index *row_number*, et à la ligne *field_number*.

10.21.6 filepro_fieldcount

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filepro_fieldcount](#)

[PHP 3, PHP 4]

[filepro_fieldcount\(\)](#) retourne le nombre de champs (ou colonnes) d'une base filePro.

Voir aussi [filepro\(\)](#).

10.21.7 filepro_rowcount

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filepro_rowcount](#)

[PHP 3, PHP 4]

[filepro_rowcount\(\)](#) retourne le nombre de lignes dans une base filePro.

Voir aussi [filepro\(\)](#).

10.22 Système de fichiers

[\[Notes en ligne\]](#)

10.22.1 basename

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [basename](#) (string *path*)
[PHP 3, PHP 4]

[basename\(\)](#) prend en paramètre le chemin complet d'un fichier et en extrait le nom du fichier. Sous Windows, les caractères (/) et antislash (\) sont utilisés comme séparateur de dossier. Sous les autres OS, seul le caractère slash (/) est utilisé.

Exemple avec basename()

```
<?php
$path = "/home/httpd/html/index.php3";
$file = basename($path); // $file est affecté avec "index.php3"
?>
```

Voir aussi: [dirname\(\)](#).

10.22.2 chgrp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chgrp](#) (string *filename*, mixed *group*)
[PHP 3, PHP 4]

[chgrp\(\)](#) essaie de changer le groupe propriétaire du fichier. Seul le superuser (root) peut changer le groupe propriétaire d'un fichier arbitrairement. Les utilisateurs classiques ne peuvent changer le groupe propriétaire d'un fichier que si l'utilisateur propriétaire du fichier est membre du groupe.

Renvoie TRUE en cas de succès, sinon renvoie FALSE.

Voir aussi [chown\(\)](#) et [chmod\(\)](#).

Note : [chgrp\(\)](#) ne fonctionne pas sous Windows.

10.22.3 chmod

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chmod](#) (string *filename*, int *mode*)
[PHP 3, PHP 4]

[chmod\(\)](#) remplace le mode du fichier *filename* par le mode *mode*.

Il est à noter que le mode *mode* est considéré comme un nombre en notation octale. Afin de vous en assurer, vous pouvez préfixer cette valeur par un zéro (*mode*):

```
chmod( "/somedir/somefile", 755 ); // notation décimale; probablement FALSE chmod( "/somedir/somefile", 0755 );
```

Renvoie TRUE en cas de succès, FALSE sinon.

Voir aussi [chown\(\)](#) et [chgrp\(\)](#).

Note : [chmod\(\)](#) ne fonctionne pas sous Windows.

10.22.4 chown

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [chown](#)(string *filename*, mixed *user*)

[PHP 3, PHP 4]

[chown\(\)](#) change le groupe propriétaire du fichier. Seul le superutilisateur (root) peut changer le propriétaire d'un fichier.

Renvoie TRUE en cas de succès, "FALSE" sinon. Note : *Sous Windows, ne fait rien et retourne TRUE.*

Voir aussi [chown\(\)](#) et [chmod\(\)](#).

Note : [chown\(\)](#) est inopérante sous Windows.

10.22.5 clearstatcache

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [clearstatcache](#)

[PHP 3, PHP 4]

L'appel à la fonction `stat` ou `lstat` est relativement couteux en terme de temps d'exécution. Pour cela, le résultat du dernier appel à l'une des fonctions de statut, (voir la liste ci-dessous), est sauvegardé pour ré-utilisation ultérieure. Si vous voulez forcer la vérification du statut d'un fichier, dans le cas où le fichier aurait pu être modifié ou aurait disparu, vous devez utiliser la fonction [clearstatcache\(\)](#) afin d'effacer de la mémoire les résultats du dernier appel à la fonction.

La valeur du cache n'est valable que pour la durée d'une requête.

Les fonctions affectées sont : [stat\(\)](#), [lstat\(\)](#), [file_exists\(\)](#), [is_writable\(\)](#), [is_readable\(\)](#), [is_executable\(\)](#), [is_file\(\)](#), [is_dir\(\)](#), [is_link\(\)](#), [filectime\(\)](#), [fileatime\(\)](#), [filemtime\(\)](#), [fileinode\(\)](#), [filegroup\(\)](#), [fileowner\(\)](#), [filesize\(\)](#), [filetype\(\)](#), et [fileperms\(\)](#).

10.22.6 copy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [copy](#)(string *source*, string *dest*)

[PHP 3, PHP 4]

[copy\(\)](#) fait une copie du fichier. Elle renvoie TRUE en cas de succès, FALSE sinon.

Exemple avec copy()

```
if ( !copy($file, $file.'.bak') ) {
    print("La copie du fichier $file n'a pas réussi...<br>\n");
}
```

Voir aussi: [rename\(\)](#).

[10.22.7 delete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [delete](#) (string *file*)

Ceci est une fausse entrée du manuel pour ceux qui recherchent en fait la fonction [unlink\(\)](#) ou [unset\(\)](#). Voir aussi: [unlink\(\)](#) pour effacer des fichiers, [unset\(\)](#) pour effacer des variables.

[10.22.8 dirname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dirname](#) (string *path*)
[PHP 3, PHP 4]

Si *path* contient le chemin d'un fichier ou dossier, [dirname\(\)](#) retournera le nom du dossier qui le contient. Sous Windows, les slash (/) et anti-slash (\) sont utilisés comme séparateurs de dossier. Dans les autres environnements, seul le slash (/) est utilisé.

Exemple avec dirname()

```
<?php
$path = "/etc/passwd";
$file = dirname($path); // $file contient "/etc"
?>
```

Voir aussi: [basename\(\)](#).

[10.22.9 diskfreespace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [diskfreespace](#) (string *directory*)
[PHP 3>= 3.0.7, PHP 4 >= 4.0b4]

[diskfreespace\(\)](#) retournera le nombre d'octets disponible sur le disque correspondant contenant le dossier *directory*.

Exemple avec diskfreespace()

```
<?php
$df = diskfreespace("/"); // $df contient le nombre d'octets libres sur "/"
?>
```

[10.22.10 fclose](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fclose](#) (int *fp*)

[PHP 3, PHP 4]

[fclose\(\)](#) ferme le fichier *fp*.

[fclose\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#) ou [fsockopen\(\)](#).

10.22.11 feof

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [feof](#)(int *fp*)

[PHP 3, PHP 4]

[feof\(\)](#) retourne TRUE si le pointeur *fp* est à la fin du fichier, ou si une erreur survient, sinon, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

10.22.12 fflush

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fflush](#)(int *fp*)

[PHP 4 >= 4.0.1]

[fflush\(\)](#) forces l'écriture de toutes les données bufferisées dans le fichier désigné par *fp*. [fflush\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

fp est un pointeur de fichier ouvert avec [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

10.22.13 fgetc

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fgetc](#)(int *fp*)

[PHP 3, PHP 4]

[fgetc\(\)](#) retourne une chaîne contenant un seul caractère, lu depuis le fichier pointé par *fp*. [fgetc\(\)](#) retourne FALSE à la fin du fichier (tout comme [feof\(\)](#)).

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

Voir aussi [fread\(\)](#), [fopen\(\)](#), [popen\(\)](#), [fsockopen\(\)](#), et [fgets\(\)](#).

10.22.14 fgetcsv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [fgetcsv](#)(int *fp*, int *length*, string *delimiter*)

[PHP 3 >= 3.0.8, PHP 4]

Identique à [fgets\(\)](#) mais [fgetcsv\(\)](#) analyse la ligne qu'il lit et recherche les champs CSV, qu'il va retourner dans un tableau les contenant. Le délimiteur de champs *delimiter* est la virgule, à moins que vous ne fournissiez un troisième argument.

fp doit être un pointeur valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

length doit être plus grand que la plus grande ligne trouvée dans un fichier CSV (en comptant les caractères

de fin de ligne).

[fgetcsv\(\)](#) retourne FALSE en cas d'erreur, ou en cas de fin du fichier.

Note : une ligne vide dans un fichier CSV sera retournée dans le tableau comme une chaîne vide, et ne sera pas traitée comme une erreur.

Exemple avec fgetcsv() une ligne vide dans un fichier csv sera retournée dans le tableau comme une chaîne vide, et ne sera pas traitée comme une erreur.

```
<?php
$row = 1;
$fp = fopen ("test.csv", "r");
while ($data = fgetcsv ($fp, 1000, ",")) {
    $num = count ($data);
    print "<p> $num champs dans la ligne $row: <br>";
    $row++;
    for ($c=0; $c<$num; $c++) {
        print $data[$c] . "<br>";
    }
}
fclose ($fp);
?>
```

10.22.15 fgets

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fgets](#) (int *fp*, int *length*)
[PHP 3, PHP 4]

[fgets\(\)](#) retourne la chaîne lue jusqu'à la longueur *length* – 1 octets, ou bien la fin du fichier, ou encore un retour chariot (le premier des trois qui sera rencontré).

Si une erreur survient, retourne FALSE.

Erreur courante :

Les programmeurs habitués à la programmation 'C' noteront que [fgets\(\)](#) ne se comporte pas comme son équivalent C lors de la rencontre de la fin du fichier.

fp doit être valide, et avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#).

Un exemple simple :

Lecture d'un fichier ligne par ligne

```
<?php
$fd = fopen ("/tmp/inputfile.txt", "r");
while (!feof($fd)) {
    $buffer = fgets($fd, 4096);
    echo $buffer;
}
fclose ($fd);
?>
```

Voir aussi [fread\(\)](#), [fopen\(\)](#), [popen\(\)](#), [fgetc\(\)](#), [fsockopen\(\)](#), et [socket_set_timeout\(\)](#).

10.22.16 fgetss

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fgetss](#) (int *fp*, int *length*, string *allowable_tags*)
[PHP 3, PHP 4]

Identique à [fgets\(\)](#), mais [fgetss\(\)](#) supprime toutes les balises HTML et PHP qu'il trouve dans le texte lu. Vous pouvez aussi préciser les balises qui seront ignorées dans le troisième paramètre, optionnel. Note : *allowable_tags* a été ajouté dans PHP 3.0.13, PHP 4B3.

Voir aussi [fgets\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), [popen\(\)](#), et [strip_tags\(\)](#).

10.22.17 file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [file](#) (string *filename*, int *use_include_path*)
[PHP 3, PHP 4]

Identique à [readfile\(\)](#), hormis le fait que [file\(\)](#) retourne le fichier dans un tableau. Chaque élément du tableau correspond à une ligne du fichier, et les retour chariots sont placés en fin de ligne.

Vous pouvez utiliser l'option et en la mettant à "1", si vous voulez rechercher aussi dans le dossier [7.1.1.14 ini.include-path](#).

```
<?php
// Lire une page web dans un tableau, et l'afficher
$fcontents = file( 'http://www.php.net' );
while ( list( $numero_ligne, $ligne ) = each( $fcontents ) ) {
echo "<B>Ligne $numero_ligne:</B> " . htmlspecialchars( $ligne ) . "<br>\n";
}
// lire une page web dans une chaîne
$fcontents = join( '', file( 'http://www.php.net' ) );
?>
```

Voir aussi [readfile\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), et [popen\(\)](#).

10.22.18 file_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [file_exists](#) (string *filename*)
[PHP 3, PHP 4]

[file_exists\(\)](#) retourne TRUE si le fichier *filename* existe, et FALSE sinon.

[file_exists\(\)](#) ne fonctionne pas sur les fichiers distants. Les fichiers doivent être accessibles par le système de fichier du serveur.

Le résultat de [file_exists\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

10.22.19 filemtime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filemtime](#) (string *filename*)
[PHP 3, PHP 4]

[fileatime\(\)](#) renvoie la date à laquelle le fichier a été accédé pour la dernière fois, ou FALSE en cas d'erreur. Le résultat de [fileatime\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.22.20 filectime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filectime](#) (string *filename*)
[PHP 3, PHP 4]

[filectime\(\)](#) renvoie l'heure à laquelle l'inode *filename* a été accédé pour la dernière fois, ou FALSE en cas d'erreur.

Le résultat de [filectime\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

[10.22.21 filegroup](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filegroup](#) (string *filename*)
[PHP 3, PHP 4]

[filegroup\(\)](#) renvoie le groupe qui possède le fichier *filename* ou FALSE en cas d'erreur. L'identifiant de groupe est retourné au format numérique, utilisez [posix_getgrgid\(\)](#) pour retrouver le nom du groupe.

Le résultat de [filegroup\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note : [filegroup\(\)](#) est inopérante sous Windows.

[10.22.22 fileinode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileinode](#) (string *filename*)
[PHP 3, PHP 4]

[fileinode\(\)](#) renvoie le numéro d'inode du fichier *filename*, ou FALSE en cas d'erreur.

Le résultat de [fileinode\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note : [fileinode\(\)](#) est inopérante sous Windows.

[10.22.23 filemtime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filemtime](#) (string *filename*)
[PHP 3, PHP 4]

[filemtime\(\)](#) renvoie la date de dernière modification du fichier *filename*, ou FALSE en cas d'erreur.

Le résultat de [filemtime\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note: [filemtime\(\)](#) retourne l'heure d'écriture des blocs données d'un fichier. Utilisez [date\(\)](#) sur ce résultat pour obtenir une date de modification humainement lisible.

10.22.24 fileowner

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileowner](#)(string *filename*)

[PHP 3, PHP 4]

[fileowner\(\)](#) renvoie le nom du possesseur du fichier *filename*, ou FALSE en cas d'erreur. L'identification du possesseur de fichier est numérique : il faut utiliser [posix_getpwuid\(\)](#) pour retrouver le nom d'utilisateur. Le résultat de [fileowner\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

Note : [fileowner\(\)](#) est inopérante sous Windows.

10.22.25 fileperms

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fileperms](#)(string *filename*)

[PHP 3, PHP 4]

[fileperms\(\)](#) renvoie les permissions affectées au fichier *filename*, ou FALSE en cas d'erreur. Le résultat de [fileperms\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

10.22.26 filesize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [filesize](#)(string *filename*)

[PHP 3, PHP 4]

[filesize\(\)](#) renvoie la taille du fichier *filename*, ou FALSE en cas d'erreur. Le résultat de [filesize\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

10.22.27 filetype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [filetype](#)(string *filename*)

[PHP 3, PHP 4]

[filetype\(\)](#) renvoie le type du fichier *filename*. Les réponses possibles sont : fifo, char, dir, block, link, file, et unknown.

[filetype\(\)](#) retourne FALSE en cas d'erreur.

Le résultat de [filetype\(\)](#) est mis en cache. Reportez vous à [clearstatcache\(\)](#) pour plus de détails.

10.22.28 flock

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [flock](#)(int *fp*, int *operation*)

[PHP 3>= 3.0.7, PHP 4]

PHP dispose d'un système complet de verrouillage de fichiers. Tous les programmes qui accèdent au fichier

doivent utiliser la même méthode de verrouillage pour qu'il soit efficace.

[flock\(\)](#) agit sur le fichier *fp* qui doit avoir été ouvert au préalable. *operation* est une des valeurs suivantes :

- Acquisition d'un verrou : *operation* = 1.
- Acquisition d'un verrou exclusif (écriture), *operation* = 2.
- Libération d'un verrou (partagé ou exclusif), *operation* = 3.
- Si vous voulez que [flock\(\)](#) ne se bloque pas durant le verrouillage, ajoutez 4 à *operation*.

[flock\(\)](#) permet de réaliser un système simple de verrous écriture / lecture, qui peut être utilisé sur n'importe quelle plateforme (Unix et Windows compris).

[flock\(\)](#) retourne TRUE en cas de succès, et FALSE sinon. (le verrou n'a pas pu être obtenu).

Sur la plus part des OS, [flock\(\)](#) est implémenté au niveau processus. Lors de l'utilisation des API d'un serveur multi-thread, comme ISAPI, vous ne pouvez pas vous fier à [flock\(\)](#) pour protéger vos fichiers contre des accès concurrents du même serveur.

10.22.29 fopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fopen](#) (string *filename*, string *mode*, int *use_include_path*)
[PHP 3, PHP 4]

Si *filename* commence par "http://" (insensible à la casse), une connexion HTTP 1.x est ouverte avec le serveur spécifié, et un pointeur sur la réponse fournie est retourné.

Attention, [fopen\(\)](#) ne gère pas les redirections, ce qui oblige à ajouter les slash " / " finaux pour indiquer un dossier.

Si *filename* commence par "ftp://" (insensible à la casse), une connexion FTP est ouverte avec le serveur spécifié, et un pointeur sur la réponse fournie est retourné. Si le serveur ne supporte pas le mode FTP passif, [fopen\(\)](#) échouera. Vous pouvez ouvrir des fichiers en lecture seulement, ou en écriture seulement (le full duplex n'est pas supporté).

Si *filename* commence par "php://stdin", "php://stdout", ou "php://stderr", le flot correspondant sera ouvert. (Cela a été introduit dans PHP 3.0.13; dans les anciennes versions, les fichiers "/dev/stdin" ou "/dev/fd/0" devaient être utilisés pour accéder à ces flots).

Si *filename* commence par n'importe quoi d'autre, PHP tentera de lire ce fichier dans le système local, et un pointeur sur le fichier ouvert sera retourné.

Si l'ouverture échoue, [fopen\(\)](#) retourne FALSE.

mode peut prendre les valeurs suivantes :

- 'r' – Ouvre en lecture seule, et place le pointeur de fichier au début du fichier.
- 'r+' – Ouvre en lecture et écriture, et place le pointeur de fichier au début du fichier.
- 'w' – Ouvre en écriture seule; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.
- 'w+' – Ouvre en lecture et écriture; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.
- 'a' – Ouvre en écriture seule; place le pointeur de fichier à la fin du fichier file. Si le fichier n'existe pas, on tente de le créer.
- 'a+' – Ouvre en lecture et écriture; place le pointeur de fichier à la fin du fichier. Si le fichier n'existe pas, on tente de le créer.

De plus, *mode* peut contenir la lettre 'b'. Cette option n'est utile que sur les systèmes qui font la différence

entre les fichiers binaires et les fichiers textes (en bref, c'est inutile sous Unix). Si il n'est pas nécessaire, il sera ignoré.

Vous pouvez utiliser le troisième paramètre optionnel pour explorer le dossier [7.1.1.14 ini.include-path](#), en le mettant à 1.

Exemple avec fopen()

```
<?php
$fp = fopen("/home/rasmus/file.txt", "r");
$fp = fopen("http://www.php.net/", "r");
$fp = fopen("ftp://user:password@example.com/", "w");
?>
```

Si vous rencontrez des problèmes en lecture ou écriture de fichier et que vous utilisez PHP en version module de serveur, n'oubliez pas que les fichiers auxquels vous accédez ne sont pas nécessairement accessibles au processus serveur.

Sous Windows, assurez vous de bien échapper les anti-slash utilisés dans le chemin du fichier, ou bien utilisez des slash.

```
<?php
$fp = fopen("c:\\data\\info.txt", "r");
?>
```

Voir aussi [fclose\(\)](#), [fsockopen\(\)](#), [socket_set_timeout\(\)](#), et [popen\(\)](#).

10.22.30 fpassthru

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fpassthru](#) (int *fp*)

[PHP 3, PHP 4]

[fpassthru\(\)](#) lit tout le reste d'un fichier jusqu'à la fin, et dirige le résultat vers la sortie standard.

Si une erreur survient, [fpassthru\(\)](#) retourne FALSE.

Le pointeur de fichier doit être valide, et doit avoir été correctement ouvert par [fopen\(\)](#), [popen\(\)](#), ou [fsockopen\(\)](#). Après lecture, [fpassthru\(\)](#) va fermer le fichier (le pointeur *fp* sera alors invalide).

Si vous voulez simplement afficher le contenu d'un fichier, il suffit d'utiliser [readfile\(\)](#), ce qui épargnera l'appel à [fopen\(\)](#).

Voir aussi [readfile\(\)](#), [fopen\(\)](#), [popen\(\)](#), et [fsockopen\(\)](#).

10.22.31 fputs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fputs](#) (int *fp*, string *str*, int *length*)

[PHP 3, PHP 4]

[fputs\(\)](#) est un alias de [fwrite\(\)](#), et lui est identique en tout point. Notez que *length* est un paramètre optionnel, et si il n'est pas spécifié, toute la chaîne est écrite.

10.22.32 fread

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [fread](#)(int *fp*, int *length*)

[PHP 3, PHP 4]

[fread\(\)](#) lit jusqu'à *length* octets dans le fichier référencé par *fp*. La lecture s'arrête lorsque *length* octets ont été lus, ou que l'on a atteint la fin du fichier, ou qu'une erreur survient (le premier des trois).

```
<?php
// Lit un fichier, et le place dans une chaîne
$filename = "/usr/local/something.txt";
$fd = fopen ($filename, "r");
$contents = fread ($fd, filesize ($filename));
fclose ($fd);
?>
```

Voir aussi [fwrite\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), [popen\(\)](#), [fgets\(\)](#), [fgetss\(\)](#), [file\(\)](#), et [fpassthru\(\)](#).

10.22.33 fscanf

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [fscanf](#)(int *handle*, string *format*, string *var1*...)

[PHP 4 >= 4.0.1]

[fscanf\(\)](#) est similaire à [sscanf\(\)](#), mais elle prend comme entrée un fichier, associé à *handle* et l'interprète en fonction du format *format*. Si seulement deux paramètres sont passés à la fonction, les valeurs parsées seront retournées sous forme de tableau. Si des arguments optionnels sont passés, la fonction retournera le nombre de valeurs assignées. Les options doivent être passées par référence.

Exemple [fscanf\(\)](#)

```
<?php
$fp = fopen ("users.txt", "r");
while ($userinfo = fscanf ($fp, "%s\t%s\t%s\n")) {
    list ($name, $profession, $countrycode) = $userinfo;
    //... traitement des données
}
fclose($fp);
?>
```

users.txt

```
janus  argonaute      gr
rodin  sculpteur       fr
som    oncle us
leonard inventeur it
```

Voir aussi [fread\(\)](#), [fgets\(\)](#), [fgetss\(\)](#), [sscanf\(\)](#), [printf\(\)](#), et [sprintf\(\)](#).

10.22.34 fseek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fseek](#)(int *fp*, int *offset*)

[PHP 3, PHP 4]

[fseek\(\)](#) modifie le curseur de position dans le fichier *fp*. La nouvelle position mesurée en octets à partir du début du fichier, est obtenue en additionnant la distance *offset* à la position *whence*. Ce paramètre peut prendre les valeurs suivantes :

- SEEK_SET – La position finale vaut *offset* octets.
- SEEK_CUR – La position finale vaut la position courante ajoutée à *offset* octets.
- SEEK_END – La position finale vaut la position courante par rapport à la fin du fichier, ajoutée de *offset*.

Si *whence* n'est pas spécifiée, il vaut par défaut SEEK_SET.

[fseek\(\)](#) retourne TRUE en cas de succès, et sinon -1. Notez que positionner le pointeur au delà de la fin du fichier n'est pas une erreur.

Ne peut pas être utilisé sur les pointeurs retournés par [fopen\(\)](#) s'ils sont au format HTTP ou FTP.

Voir aussi [tello\(\)](#) et [rewind\(\)](#).

10.22.35 fstat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [fstat](#)(int *fp*)

[PHP 4 >= 4.0RC1]

[fstat\(\)](#) rassemble les informations sur le fichier dont on connaît le pointeur *fp*. [fstat\(\)](#) est similaire à la fonction [stat\(\)](#), hormis le fait qu'elle utilise un pointeur de fichier, au lieu d'un nom de fichier.

[fstat\(\)](#) retourne un tableau avec les éléments suivants :

1. volume
2. inode
3. mode de protection du inode
4. nombre de liens
5. id de l'utilisateur propriétaire
6. id du groupe propriétaire
7. type du volume de l' inode *
8. taille en octets
9. date du dernier accès
10. date de la dernière modification
11. date du dernier changement
12. taille de bloc du système pour les entrées/sorties *
13. Nombre de blocs alloués

* – uniquement sur les systèmes qui supporte le type st_blksize typeBles autres systèmes (i.e. Windows) retourne -1

Les résultats de [fstat\(\)](#) sont mis en cache . Reportez vous à la fonction [clearstatcache\(\)](#) pour plus de détails.

10.22.36 ftell

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftell](#)(int *fp*)

[PHP 3, PHP 4]

[ftell\(\)](#) retourne la position courante du pointeur dans le fichier repéré par le pointeur *fp*, i.e., son offset.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#) ou [popen\(\)](#).

Voir aussi [fopen\(\)](#), [popen\(\)](#), [fseek\(\)](#) et [rewind\(\)](#).

10.22.37 ftruncate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftruncate](#)(int *fp*, int *size*)

[PHP 4 >= 4.0RC1]

[ftruncate\(\)](#) prend le pointeur de fichier *fp* et le tronque à la taille de *size*. La fonction retourne TRUE en cas de succès, et FALSE sinon.

10.22.38 fwrite

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fwrite](#)(int *fp*, string *string*, int *length*)

[PHP 3, PHP 4]

[fwrite\(\)](#) écrit le contenu de la chaîne *string* dans le fichier pointé par *fp*. Si la longueur *length* est fournie, l'écriture s'arrêtera après *length* octets, ou à la fin de la chaîne (le premier des deux).

Notez que si *length* est fourni, alors l'option de configuration [7.1.1.18 ini.magic-quotes-runtime](#) sera ignorée, et les slash seront conservés.

Voir aussi [fread\(\)](#), [fopen\(\)](#), [fsockopen\(\)](#), [popen\(\)](#), et [fputs\(\)](#).

10.22.39 set_file_buffer

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [set_file_buffer](#)(int *fp*, int *buffer*)

[PHP 3 >= 3.0.8, PHP 4 >= 4.0.1]

L'écriture de fichier avec [fwrite\(\)](#) utilise normalement un buffer de 8K. Cela signifie que si deux processus essaie d'écrire dans le même fichier, ils font une pause tous les 8ko pour laisser le temps à l'autre d'écrire à son tour. [set_file_buffer\(\)](#) permet de modifier la taille du buffer de sortie pour le pointeur de fichier *fp* à *buffer* octets. Si *buffer* vaut 0, l'écriture se fera sans buffer. Cela force l'écriture de toutes les données par un processus avant que les autres puisse accéder au fichier.

La fonction retourne 0 en cas de succès, ou EOF si la requête ne peut pas être honorée.

L'exemple suivant montre comment utiliser la fonction [set_file_buffer\(\)](#) pour créer un fichier sans buffer.

Exemple avec set_file_buffer()

```
<?php
```

```

$fp=fopen($file, "w");
if($fp){
    set_file_buffer($fp, 0);
    fputs($fp, $output);
    fclose($fp);
}
?>

```

Voir aussi [fopen\(\)](#), [fwrite\(\)](#).

10.22.40 is_dir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_dir](#)(string *filename*)
[PHP 3, PHP 4]

[is_dir\(\)](#) retourne TRUE si *filename* existe et est un dossier.

Le résultat de [is_dir\(\)](#) est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_file\(\)](#) et [is_link\(\)](#).

10.22.41 is_executable

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_executable](#)(string *filename*)
[PHP 3, PHP 4]

[is_executable\(\)](#) retourne TRUE si *filename* existe et est exécutable.

Le résultat de [is_executable\(\)](#) est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_file\(\)](#) et [is_link\(\)](#).

10.22.42 is_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_file](#)(string *filename*)
[PHP 3, PHP 4]

[is_file\(\)](#) retourne TRUE si *filename* existe et est un fichier (et pas un dossier).

Le résultat de [is_file\(\)](#) est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_dir\(\)](#) et [is_link\(\)](#).

10.22.43 is_link

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_link](#)(string *filename*)
[PHP 3, PHP 4]

[is_link\(\)](#) retourne TRUE si *filename* existe et est un lien symbolique.

Le résultat de [is_link\(\)](#) est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_dir\(\)](#) et [is_file\(\)](#).

Note : [is_link\(\)](#) est inopérante sous Windows.

10.22.44 [is_readable](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_readable](#) (string *filename*)

[PHP 3, PHP 4]

[is_readable\(\)](#) retourne TRUE si *filename* existe et est accessible en lecture.

N'oubliez pas que PHP accède aux fichiers avec les mêmes autorisations que l'utilisateur qui fait tourner le serveur web (souvent, c'est 'nobody', personne).

Le résultat de [is_readable\(\)](#) est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_writable\(\)](#).

10.22.45 [is_writable](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_writable](#) (string *filename*)

[PHP 4 >= 4.0b2]

[is_writable\(\)](#) retourne TRUE si *filename* existe et est accessible en lecture.

N'oubliez pas que PHP accède aux fichiers avec les mêmes autorisations que l'utilisateur qui fait tourner le serveur web (souvent, c'est 'nobody', personne).

Le résultat de [is_writable\(\)](#) est mis en cache. Voir la fonction [clearstatcache\(\)](#) pour plus de détails.

Voir aussi [is_readable\(\)](#).

10.22.46 [is_uploaded_file](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_uploaded_file](#) (string *filename*)

[PHP 3 >= 3.0.17, PHP 4 >= 4.0.3]

[is_uploaded_file\(\)](#) est disponible à partir des versions PHP 3.0.16 et 4.0.2.

[is_uploaded_file\(\)](#) retourne TRUE si le fichier *filename* a été téléchargé par HTTP POST. Cela est très utile pour vous assurer qu'un utilisateur n'essaie pas d'accéder intentionnellement à un fichier auquel il n'a pas droit (comme `/etc/passwd`).

Ce type de vérification est spécialement important si il est possible que les fichiers téléchargés révèlent leur contenu à l'utilisateur, ou même aux utilisateurs du même système.

Voir aussi [move_uploaded_file\(\)](#), et la section [8.4 Gestion des chargements de fichier](#) pour un exemple simple.

10.22.47 [link](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [link](#) (string *target*, string *link*)

[PHP 3, PHP 4]

[link\(\)](#) crée un lien.

Voir aussi [symlink\(\)](#) pour créer des liens symboliques et [readlink\(\)](#) avec [linkinfo\(\)](#).

Note : [link\(\)](#) est inopérante sous Windows.

10.22.48 linkinfo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [linkinfo](#)(string *path*)

[PHP 3, PHP 4]

[linkinfo\(\)](#) renvoie les informations à propos d'un lien le champs st_dev de la structure d'information d' UNIX (comme en langage C). [linkinfo\(\)](#) sert à vérifier si un lien (repéré par *path*) existe (en utilisant la même méthode que la macro S_ISLNK de stat.h). [linkinfo\(\)](#) retourne FALSE en cas d'erreur.

Voir aussi [symlink\(\)](#), [link\(\)](#), et [readlink\(\)](#).

Note : [linkinfo\(\)](#) est inopérante sous Windows.

10.22.49 mkdir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mkdir](#)(string *pathname*, int *mode*)

[PHP 3, PHP 4]

[mkdir\(\)](#) tente de créer un dossier dans le chemin *pathname*.

Notez que vous aurez à préciser le mode en base octale, ce qui signifie que vous aurez probablement un 0 comme premier chiffre :

```
<?php
mkdir ( "/chemin/de/mon/dossier", 0700 );
?>
```

[mkdir\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

Voir aussi [rmdir\(\)](#).

10.22.50 move_uploaded_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [move_uploaded_file](#)(string *filename*, string *destination*)

[PHP 4 >= 4.0.3]

[move_uploaded_file\(\)](#) est disponible à partir des versions PHP 3.0.16 et 4.0.2.

[move_uploaded_file\(\)](#) s'assure que le fichier *filename* est un fichier téléchargé par HTTP POST. Si le fichier est valide, il est déplacé jusqu'à *destination*.

Si *filename* n'est pas valide, rien ne se passe, et [move_uploaded_file\(\)](#) retournera FALSE.

Si *filename* est un fichier téléchargé, mais que pour une raison quelconque, il ne peut être déplacé, rien ne se passe, et [move_uploaded_file\(\)](#) retourne FALSE. De plus, une alerte sera affichée.

Ce type de vérification est spécialement important s'il est possible que les fichiers téléchargés révèlent leur contenu à l'utilisateur, ou même aux utilisateurs du même système.

Voir aussi [move_uploaded_file\(\)](#), et la section [8.4 Gestion des chargements de fichier](#) pour un exemple

simple.

[10.22.51 pclose](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pclose](#)(int *fp*)

[PHP 3, PHP 4]

[pclose\(\)](#) ferme un processus de pointeur de fichier ouvert par [popen\(\)](#).

Le pointeur de fichier doit être valide, et avoir été ouvert correctement par [popen\(\)](#).

[pclose\(\)](#) retourne le statut final du processus exécuté.

Voir aussi [popen\(\)](#).

[10.22.52 popen](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [popen](#)(string *commande*, string *mode*)

[PHP 3, PHP 4]

[popen\(\)](#) ouvre un processus fils en faisant un fork de la commande.

[popen\(\)](#) retourne un pointeur de fichier identique à celui retourné par [fopen\(\)](#), hormis le fait qu'il sera unidirectionnel (lecture seule, ou écriture seule), et doit être terminé par [pclose\(\)](#). Ce pointeur peut être utilisé avec [fgets\(\)](#), [fgetss\(\)](#), et [fputs\(\)](#).

Si une erreur survient, retourne FALSE.

```
<?php
$fp = popen ("/bin/ls", "r");
?>
```

Voir aussi [pclose\(\)](#).

[10.22.53 readfile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [readfile](#)(string *filename*, int *use_include_path*)

[PHP 3, PHP 4]

[readfile\(\)](#) lit le fichier *filename* et l'envoi à la sortie standard.

[readfile\(\)](#) retourne le nombre d'octets lus depuis le fichier. Si une erreur survient, retourne FALSE.

Si *filename* commence par "http://" (insensible à la casse), une connexion HTTP 1.0 sera ouverte avec le serveur spécifié, et le texte de la réponse sera affiché sur la sortie standard.

ATTENTION : PHP ne gère pas les redirections, donc il faut penser à ajouter un "/" final pour les dossiers.

Si *filename* commence par "ftp://" (insensible à la casse), une connexion FTP est ouverte avec l'hôte spécifié et la réponse du serveur est affichée. Si le serveur ne supporte les connexions passives, la requête échouera.

Si *filename* ne commence par aucun des cas précédents, le fichier sera ouvert sur l'hôte local, et envoyé à la sortie standard.

Vous pouvez utiliser le deuxième paramètre optionnel pour explorer le dossier [7.1.1.14 ini.include-path](#), en passant la valeur de 1.

Voir aussi [fpassthru\(\)](#), [file\(\)](#), [fopen\(\)](#), [include\(\)](#), [require\(\)](#), et [virtual\(\)](#).

[10.22.54 readlink](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [readlink](#) (string *path*)
[PHP 3, PHP 4]

[readlink\(\)](#) fait la même chose que la fonction `readlink` en C : elle retourne le contenu du lien symbolique repéré par *path*, ou FALSE en cas d'erreur.

Voir aussi [symlink\(\)](#), [readlink\(\)](#) et [linkinfo\(\)](#).

Note : [readlink\(\)](#) est inopérante sous Windows.

[10.22.55 rename](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rename](#) (string *oldname*, string *newname*)
[PHP 3, PHP 4]

[rename\(\)](#) tente de renommer le fichier *oldname* en *newname*.

[rename\(\)](#) retourne TRUE en cas de succès et FALSE sinon.

[10.22.56 rewind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rewind](#) (int *fp*)
[PHP 3, PHP 4]

[rewind\(\)](#) replace le pointeur du fichier *fp* au début.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été correctement ouvert par [fopen\(\)](#).

Voir aussi [fseek\(\)](#) et [ftell\(\)](#).

[10.22.57 rmdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rmdir](#) (string *dirname*)
[PHP 3, PHP 4]

[rmdir\(\)](#) tente d'effacer le dossier dont le chemin est *dirname*. Le dossier doit être vide, et le script doit avoir les autorisations adéquates.

Si une erreur survient, retourne FALSE.

Voir aussi [mkdir\(\)](#).

10.22.58 stat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [stat](#)(string *filename*)

[PHP 3, PHP 4]

[stat\(\)](#) renvoie les informations à propos du fichier filename.

[stat\(\)](#) retourne un tableau avec les éléments suivants :

1. volume
2. inode
3. droits d'accès au fichier (mode de protection du inode). A convertir en octal. Voir aussi [@xref{function.fperms , , fperms\(\)}](#).
4. nombre de liens
5. id de l'utilisateur propriétaire
6. id du groupe propriétaire
7. type du volume de l' inode *
8. taille en octets
9. date du dernier accès
10. date de la dernière modification
11. date du dernier changement
12. taille de bloc du système pour les entrées/sorties *
13. nombre de blocs alloués

* – uniquement sur les systèmes qui supporte le type st_blksize type les autres systèmes (i.e. Windows) retourne -1.

Les résultats de [stat\(\)](#) sont mis en cache . Reportez vous à la fonction [clearstatcache\(\)](#) pour plus de détails.

10.22.59 lstat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [lstat](#)(string *filename*)

[PHP 3>= 3.0.4, PHP 4]

[lstat\(\)](#) est identique à [stat\(\)](#) mais elle accepte aussi un lien symbolique comme argument.

[lstat\(\)](#) retourne un tableau avec les éléments suivants :

1. volume
2. inode
3. droits d'accès au fichier (mode de protection du inode). A convertir en octal. Voir aussi [@xref{function.fperms , , fperms\(\)}](#).
4. nombre de liens
5. id de l'utilisateur propriétaire
6. id du groupe propriétaire
7. type du volume de l' inode *
8. taille en octets
9. date du dernier accès
10. date de la dernière modification
11. date du dernier changement

- 12. taille de bloc du système pour les entrées/sorties *
- 13. nombre de blocs alloués

* – uniquement sur les systèmes qui supporte le type `st_blksize` type les autres systèmes (i.e. Windows) retourne -1.

Les résultats de [lstat\(\)](#) sont mis en cache . Reportez vous à la fonction [clearstatcache\(\)](#) pour plus de détails.

[10.22.60 realpath](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [realpath](#) (string *path*)
[PHP 4 >= 4.0b4]

[realpath\(\)](#) résoud tous les liens symboliques, et remplace toutes les références `'./'`, `'../'` et `'/'` de *path* puis retourne le chemin canonique absolu ainsi trouvé. Le résultat ne contient aucun lien symbolique, `'./'` ou `'../'`.

Exemple realpath()

```
<?php
$real_path = realpath ("../..../index.php");
?>
```

[10.22.61 symlink](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [symlink](#) (string *target*, string *link*)
[PHP 3, PHP 4]

[symlink\(\)](#) crée un lien symbolique pour l'objet *target* avec le nom de *link*.

Voir aussi [link\(\)](#) pour créer des liens durs, et [readlink\(\)](#) ainsi que [linkinfo\(\)](#).

Note : [symlink\(\)](#) est inopérante sous Windows.

[10.22.62 tempnam](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [tempnam](#) (string *dir*, string *prefix*)
[PHP 3, PHP 4]

[tempnam\(\)](#) crée un fichier temporaire unique dans le dossier *dir*. Si le dossier n'existe pas, [tempnam\(\)](#) va générer un nom de fichier dans le dossier temporaire du système.

Le comportement de [tempnam\(\)](#) dépend du système. Sous Windows, la variable d'environnement TMP sera prioritaire par rapport au paramètre *dir*; sous Linux la variable d'environnement TMPDIR aura la priorité, tandis que l'OS SVR4 utilisera toujours le paramètre *dir* si le dossier n'existe pas. Reportez vous à la documentation de votre système([tempnam\(3\)](#)).

[tempnam\(\)](#) retourne le nom du fichier temporaire, ou la chaîne NULL en cas d'échec.

Exemple avec tempnam()

```
<?php
$tmpfname = tempnam ( "/tmp", "FOO" );
?>
```

10.22.63 tmpfile

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [tmpfile](#)(void)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[tmpfile\(\)](#) crée un fichier temporaire avec un nom unique, ouvert en écriture, et retourne un pointeur de fichier, identique à ceux retourné par [fopen\(\)](#). Ce fichier sera automatiquement effacé lorsqu'il sera fermé (avec [fclose\(\)](#)), ou lorsque le fichier sera terminé.

Pour plus de détails, consultez votre documentation système sur la fonction tmpfile(3), et sur ``stdio.h'`.

Voir aussi [tempnam\(\)](#).

10.22.64 touch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [touch](#)(string *filename*, int *time*)

[PHP 3, PHP 4]

[touch\(\)](#) tente de forcer la date de modification du fichier nommé *filename* à la date de *time*. Si *time* est omis, c'est l'heure courante qui est utilisée.

Si le fichier n'existe pas, il est créé.

[touch\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

Exemple avec touch()

```
if ( touch($NomDeFichier) ) {
print "La date de modification de $NomDeFichier a été fixée à maintenant";
} else {
print "Désolé, il est impossible de changer la date de modification de $NomDeFichier";
}
```

10.22.65 umask

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [umask](#)(int *mask*)

[PHP 3, PHP 4]

[umask\(\)](#) change le umask de PHP : `mask & 0777` et retourne le vieux umask. Lorsque PHP est utilisé comme module de serveur, le umask reprend sa valeur à la fin de chaque script.

[umask\(\)](#) appelé sans argument retourne simplement le umask courant.

10.22.66 unlink

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [unlink](#)(string *filename*)

[PHP 3, PHP 4]

[unlink\(\)](#) efface *filename*. Identique à la fonction Unix C unlink().

[unlink\(\)](#) retourne FALSE en cas d'échec.

Voir aussi [rmdir\(\)](#) pour supprimer des dossiers.

Note : [unlink\(\)](#) peut ne pas fonctionner sous Windows.

10.23 FTP

[\[Notes en ligne\]](#)

FTP : File Transfer Protocol (Protocole de transfert de fichiers).

Les constantes suivantes sont définies dans le module FTP : FTP_ASCII et FTP_BINARY.

Exemple de connexion FTP

```

<?php
// création de la connexion
$conn_id = ftp_connect("$ftp_server");
// authentification avec nom de compte et mot de passe
$login_result = ftp_login($conn_id, "$ftp_user_name", "$ftp_user_pass");
// vérification de la connexion
if ((!$conn_id) || (!$login_result)) {
    echo "La connexion FTP a échoué!";
    echo "Tentative de connexion à $ftp_server avec $user";
    die;
} else {
    echo "Connecté à $ftp_server, avec $user";
}
// téléchargement d'un fichier
$upload = ftp_put($conn_id, "$destination_file", "$source_file", FTP_BINARY);
// Vérification de téléchargement
if (!$upload) {
    echo "Le téléchargement Ftp a échoué!";
} else {
    echo "Téléchargement de $source_file sur $ftp_server en $destination_file";
}
// fermeture de la connexion FTP.
ftp_quit($conn_id);
?>

```

10.23.1 ftp_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_connect](#)(string *host*, int *port*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_connect\(\)](#) retourne un flot **FTP** en cas de succès, et FALSE sinon.

[ftp_connect\(\)](#) ouvre une connexion FTP avec l'hôte *host*. Le paramètre *port* spécifie le port de connexion. Si il est omis, le port 21 sera utilisé.

[10.23.2 ftp_login](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_login](#)(int *ftp_stream*, string *username*, string *password*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_login\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.
[ftp_login\(\)](#) authentifie le flot FTP.

[10.23.3 ftp_pwd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_pwd](#)(int *ftp_stream*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_pwd\(\)](#) retourne le nom du dossier courant, ou FALSE en cas d'erreur.

[10.23.4 ftp_cdup](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_cdup](#)(int *ftp_stream*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_cdup\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.
[ftp_cdup\(\)](#) change de dossier, et passe au dossier parent.

[10.23.5 ftp_chdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_chdir](#)(int *ftp_stream*, string *directory*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_chdir\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.
[ftp_chdir\(\)](#) change le dossier courant en *directory*.

[10.23.6 ftp_mkdir](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ftp_mkdir](#)(int *ftp_stream*, string *directory*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_mkdir\(\)](#) retourne le nom du dossier ainsi créé en cas de succès, et FALSE sinon.
[ftp_mkdir\(\)](#) crée le dossier nommé *directory*.

10.23.7 ftp_rmdir

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_rmdir](#)(int *ftp_stream*, string *directory*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_rmdir\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_rmdir\(\)](#) efface le dossier *directory*.

10.23.8 ftp_nlist

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_nlist](#)(int *ftp_stream*, string *directory*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_nlist\(\)](#) retourne un tableau de nom de fichiers en cas de succès, et FALSE sinon.

10.23.9 ftp_rawlist

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_rawlist](#)(int *ftp_stream*, string *directory*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_rawlist\(\)](#) exécute la commande FTP LIST, et retourne le résultat dans un tableau. Chaque élément du tableau correspond à une ligne du résultat de la commande. Le résultat n'est pas analysé, et est retourné brut. L'identifiant de système retourné par [ftp_systype\(\)](#) sera utile pour déterminer la façon d'interpréter le résultat.

10.23.10 ftp_systype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_systype](#)(int *ftp_stream*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_systype\(\)](#) retourne le type de serveur, ou FALSE en cas d'erreur.

10.23.11 ftp_pasv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_pasv](#)(int *ftp_stream*, int *pasv*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_pasv\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_pasv\(\)](#) active le mode passif si *pasv* est à TRUE (et le désactive si *pasv* est à FALSE). En mode passif, les données de connexion sont initiées par le client, plutôt que par le serveur.

[10.23.12 ftp_get](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_get](#)(int *ftp_stream*, string *local_file*, string *remote_file*, int *mode*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_get\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_get\(\)](#) télécharge le fichier *remote_file* depuis le serveur FTP, et le sauve dans le fichier local *local_file*.

Le mode de transfert *mode* spécifié doit être soit FTP_ASCII ou FTP_BINARY.

[10.23.13 ftp_fget](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_fget](#)(int *ftp_stream*, int *fp*, string *remote_file*, int *mode*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_fget\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_fget\(\)](#) télécharge le fichier *remote_file* depuis le serveur FTP, et l'écrit dans le fichier identifié par *fp*. Le

mode de transfert *mode* spécifié doit être FTP_ASCII ou FTP_BINARY.

[10.23.14 ftp_put](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_put](#)(int *ftp_stream*, string *remote_file*, string *local_file*, int *mode*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_put\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_put\(\)](#) enregistre le fichier *local_file* sur le serveur FTP, sous le nom de *remote_file*. Le mode de transfert

mode spécifié doit être FTP_ASCII ou FTP_BINARY.

[10.23.15 ftp_fput](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_fput](#)(int *ftp_stream*, string *remote_file*, int *fp*, int *mode*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_fput\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_fput\(\)](#) charge les données issues du fichier identifié par *fp* jusqu'à la fin du fichier. Le résultat est stocké

dans le fichier *remote_file* sur le serveur FTP. Le mode de transfert *mode* spécifié doit être FTP_ASCII ou FTP_BINARY.

[10.23.16 ftp_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_size](#)(int *ftp_stream*, string *remote_file*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_size\(\)](#) retourne la taille du fichier en cas de succès, et FALSE sinon.

[ftp_size\(\)](#) retourne la taille d'un fichier sur un serveur FTP. Si une erreur survient, ou que le fichier n'existe pas, la valeur -1 est retournée. Certains serveurs FTP ne supportent pas cette fonction.

[10.23.17 ftp_mdtm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_mdtm](#)(int *ftp_stream*, string *remote_file*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_mdtm\(\)](#) retourne un UNIX timestamp en cas de succès, et FALSE sinon.

[ftp_mdtm\(\)](#) lit la date de dernière modification d'un fichier et retourne le UNIX timestamp. Si une erreur survient, ou si le fichier n'existe pas, la valeur -1 est retournée. Certains serveurs FTP ne supportent pas cette fonction.

[10.23.18 ftp_rename](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_rename](#)(int *ftp_stream*, string *from*, string *to*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_rename\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_rename\(\)](#) renomme le fichier *from* en *to*.

[10.23.19 ftp_delete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_delete](#)(int *ftp_stream*, string *path*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_delete\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_delete\(\)](#) efface le fichier *path* sur un serveur FTP.

[10.23.20 ftp_site](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_site](#)(int *ftp_stream*, string *cmd*)

[PHP 3>= 3.0.15, PHP 4 >= 4.0RC1]

[ftp_site\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ftp_site\(\)](#) envoie la commande *cmd* au serveur FTP. Les commandes SITE ne sont pas normalisées, et peuvent varier d'un serveur à l'autre. Elles permettent de gérer notamment les permissions de fichier, et les groupes.

10.23.21 ftp_quit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ftp_quit](#) (int *ftp_stream*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[ftp_connect\(\)](#) ferme la connexion *ftp_stream*.

10.24 Fonctions

[\[Notes en ligne\]](#)

Ces fonctions effectuent les manipulations liées à la gestion des fonctions.

10.24.1 call_user_func

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [call_user_func](#) (string *function_name* , mixed *parameter* , mixed ...)

[PHP 3>= 3.0.3, PHP 4]

Appelle la fonction utilisateur *function_name*, et lui passe les paramètres *parameter*. Par exemple :

```
function barbier ($type) {
    print "Vous vouliez une coupe $type, pas de problème";
}
call_user_func ('barbier', "iroquois");
call_user_func ('barbier', "militaire");
call_user_func ('barbier', "au bol");
```

10.24.2 create_function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [create_function](#) (string *args*, string *code*)

[PHP 4 >= 4.0.1]

Cette fonction crée une fonction anonyme, à partir des paramètres passés, et retourne un nom unique.

Généralement, les arguments *args* sont présentés sous la forme d'une chaîne à guillemets simples, et la même recommandation vaut pour *code*. La raison de l'utilisation des guillemets simples est de protéger les noms de variables du remplacement par leur valeur. Si vous utilisez les guillemets doubles, n'oubliez pas d'échapper les noms de variables (i.e. `\$avar`).

Vous pouvez utiliser cette fonction pour (par exemple) créer une fonction à partir d'informations récoltés durant l'exécution.

Creation d'une fonction anonmye avec [create_function\(\)](#)

```
<?php
$newfunc = create_function('$a,$b','return "ln($a) + ln($b) = ".log($a * $b);');
echo "Nouvelle fonction anonyme : $newfunc\n";
echo $newfunc(2,M_E)."\n";
```

```
// affichera :
// Nouvelle fonction anonyme : lambda_1
// ln(2) + ln(2.718281828459) = 1.6931471805599
?>
```

Ou, pour pouvoir appliquer une fonction générique à une liste d'arguments.

Traitement générique par fonction avec `create_function()`

```
<?php
function process($var1, $var2, $farr) {
for ($f=0; $f < count($farr); $f++)
echo $farr[$f]($var1,$var2)."\n";
}
// creation d'une série de fonction mathématiques
$f1 = 'if ($a>=0) {return "b*a^2 = ".$b*sqrt($a);} else {return FALSE;}';
$f2 = "return \"min(b^2+a, a^2,b) = \".min(\"$a*$a+$b,$b*$b+$a)\"";
$f3 = 'if ($a> 0 && $b != 0) {return "ln(a)/b = ".log($a)/$b;} else {return FALSE;}';
$farr = array(
create_function('$x,$y', 'return "un peu de trigo : ".(sin($x) + $x*cos($y));'),
create_function('$x,$y', 'return "une hypoténuse: ".sqrt($x*$x + $y*$y);'),
create_function('$a,$b', $f1),
create_function('$a,$b', $f2),
create_function('$a,$b', $f3)
);
echo "\nUtilisation de la première liste de fonctions anonymes\n";
echo "paramètres: 2.3445, M_PI\n";
process(2.3445, M_PI, $farr);
// Maintenant une liste de fonction sur chaîne de caractères
$garr = array(
create_function('$b,$a','if (strncmp($a,$b,3) == 0) return "*** \"$a\" ' .
'et \"$b\""\n** Ces chaînes de ressemblent!! (regarde les trois premiers caractères)";'),
create_function('$a,$b','; return "CRCs: ".crc32($a)." , ".crc32($b);'),
create_function('$a,$b','; return "similarité(a,b) = ".similar_text($a,$b,&$p).("($p%)");')
);
echo "\nUtilisation de la secondes liste de fonctions anonymes\n";
process("Twas brillling and the slithy toves", "Twas the night", $garr);
?>
```

Et lorsque vous utilisez le code ci dessus, l'affichage va être

```
Utilisation de la première liste de fonctions anonymes
paramètres: 2.3445, M_PI
Un peu de trigo: -1.6291725057799
Une hypoténuse: 3.9199852871011
b*a^2 = 4.8103313314525
min(b^2+a, a^2,b) = 8.6382729035898
ln(a/b) = 0.27122299212594
Utilisation de la seconde liste de fonctions anonymes
** "Twas the night" et "Twas brillling and the slithy toves"
** Ces chaînes de ressemblent!! (regarde les trois premiers caractères)
CRCs: -725381282 , 1908338681
similarité(a,b) = 11(45.833333333333%)
```

Mais l'utilisation la plus courante des fonctions lambda est la fonction de callback, par exemple lors de l'utilisation de [array_walk\(\)](#) ou [usort\(\)](#)

Utilisation de fonctions anonymes comme fonction de callback

```
<?php
$av = array("la ", "une ", "cette ", "une certaine ");
```

```

array_walk($av, create_function('&$v,$k','$v = $v."maison";'));
print_r($av); // En PHP 3 utilisez var\_dump\(\)
// affiche:
// Array
// (
//     [0] => la maison
//     [1] => une maison
//     [2] => cette maison
//     [3] => une certaine maison
// )
// un tableau de chaîne classé par taille
$sv = array("petite","moyenne","tres longue","vraiment tres longue");
print_r($sv);
// affiche:
// Array
// (
//     [0] => petite
//     [1] => moyenne
//     [2] => tres longue
//     [3] => vraiment tres longue
// )
// Tri par ordre de taille décroissant
usort($sv, create_function('$a,$b','return strlen($b) - strlen($a);'));
print_r($sv);
// outputs:
// Array
// (
//     [0] => vraiment tres longue
//     [1] => tres longue
//     [2] => moyenne
//     [3] => petite
// )
?>

```

[10.24.3 func_get_arg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [func_get_arg](#)(int *arg_num*)

[PHP 4 >= 4.0b4]

Retourne l'argument à la position *arg_num* dans la liste d'argument d'une fonction utilisateur. Les arguments sont comptés en commençant à zéro. [func_get_arg\(\)](#) générera une alerte si elle est appelée hors d'une fonction.

Si *arg_num* est plus grand que le nombre d'arguments passés, une alerte est générée et la fonction retourne FALSE.

```

<?php
function foo() {
    $numargs = func_num_args();
    echo "Nombre d'arguments: $numargs<br>\n";
    if ($numargs >= 2) {
        echo "Le second argument est: " . func_get_arg(1) . "<br>\n";
    }
}
foo(1, 2, 3);
?>

```

[func_get_arg\(\)](#) peut être utilisé conjointement à [func_num_args\(\)](#) et [func_get_args\(\)](#) pour permettre aux fonctions utilisateurs d'accepter un nombre variable d'arguments.

Note : Cette fonction a été ajoutée dans PHP 4.

10.24.4 func_get_args

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [func_get_args](#)(void)

[PHP 4 >= 4.0b4]

Retourne un tableau dont les éléments correspondent aux éléments de la liste d'arguments de la fonction. [func_get_args\(\)](#) générera une alerte si elle est appelée hors d'une fonction.

```
<?php
function foo() {
    $numargs = func_num_args();
    echo "Nombre d'arguments: $numargs<br>\n";
    if ($numargs >= 2) {
        echo "Le second argument est: " . func_get_arg (1) . "<br>\n";
    }
    $arg_list = func_get_args();
    for ($i = 0; $i < $numargs; $i++) {
        echo "L'argument $i est: " . $arg_list[$i] . "<br>\n";
    }
}
foo (1, 2, 3);
?>
```

[func_get_arg\(\)](#) peut être utilisé conjointement à [func_num_args\(\)](#) et [func_get_args\(\)](#) pour permettre aux fonctions utilisateurs d'accepter un nombre variable d'arguments.

Note : Cette fonction a été ajoutée dans PHP 4.

10.24.5 func_num_args

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [func_num_args](#)(void)

[PHP 4 >= 4.0b4]

Retourne le nombre d'arguments passé à la fonction utilisateur courante. [func_num_args\(\)](#) générera une alerte si elle est appelée hors d'une fonction.

```
<?php
function foo() {
    $numargs = func_num_args();
    echo "Nombre d'arguments: $numargs\n";
}
foo (1, 2, 3);    // affiche 'Nombre d'arguments: 3'
?>
```

[func_get_arg\(\)](#) peut être utilisé conjointement à [func_num_args\(\)](#) et [func_get_args\(\)](#) pour permettre aux

fonctions utilisateurs d'accepter un nombre variable d'arguments.

Note : Cette fonction a été ajoutée dans PHP 4.

10.24.6 function_exists

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [function_exists](#)(string *function_name*)

[PHP 3>= 3.0.7, PHP 4]

Vérifie la liste des fonctions définies par l'utilisateur, et retourne TRUE si *function_name* y est trouvé, FALSE sinon.

```
if (function_exists('imap_open')) {
echo "Les fonctions IMAP sont disponibles.<br>\n";
} else {
echo "Les fonctions IMAP ne sont pas disponibles.<br>\n";
}
```

Notez qu'une fonction peut exister, même si elle est indisponible, à cause de la configuration ou des options de compilation.

Voir aussi [method_exists\(\)](#).

10.24.7 register_shutdown_function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [register_shutdown_function](#)(string *func*)

[PHP 3>= 3.0.4, PHP 4]

Enregistre la fonction *func* pour exécution à l'extinction du script.

Erreur courante :

Etant donné qu'aucun affichage n'est possible depuis cette fonction, vous ne pouvez pas mettre d'informations de débogage par print ou echo ici!

10.25 GNU Gettext

[\[Notes en ligne\]](#)

Les fonctions gettext implémente l'API NLS (Native Language Support) qui peut servir à internationaliser vos scripts PHP. Lisez la documentation GNU pour plus d'explications sur ces fonctions.

10.25.1 bindtextdomain

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bindtextdomain](#)(string *domain*, string *directory*)

[PHP 3>= 3.0.7, PHP 4]

[bindtextdomain\(\)](#) fixe le chemin du domaine *domain* à *directory*.

10.25.2 dcgettext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dcgettext](#) (string *domain*, string *message*, int *category*)
[PHP 3>= 3.0.7, PHP 4]

[dcgettext\(\)](#) permet de remplacer le domaine courant lors de la recherche d'un message. Elle permet aussi de spécifier une catégorie.

10.25.3 dgettext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dgettext](#) (string *domain*, string *message*)
[PHP 3>= 3.0.7, PHP 4]

[dgettext\(\)](#) remplace le domaine courant.

10.25.4 gettext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gettext](#) (string *message*)
[PHP 3>= 3.0.7, PHP 4]

[gettext\(\)](#) retourne une chaîne traduite, si elle en a trouvé une dans la table de traduction, ou bien le message *message*, s'il n'a pas été trouvé. Vous pouvez utiliser le caractère souligné (_) comme alias de cette fonction.

Vérification gettext()

```
<?php
// Choix l'allemand
putenv("LANG=de");
// Spécifie la localisation des tables de traduction
bindtextdomain("myPHPApp", "./locale");
// Choisi le domaine
textdomain("myPHPApp");
// Affiche un message de test
print (gettext ("Bienvenue sur mon application PHP"));
?>
```

10.25.5 textdomain

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [textdomain](#) (string *text_domain*)
[PHP 3>= 3.0.7, PHP 4]

[textdomain\(\)](#) fixe le domaine à utiliser lors de recherche avec [gettext\(\)](#). Ce domaine dépend généralement de l'application. Le domaine par défaut précédent est retourné. Appelez cette fonction sans paramètre pour avoir la valeur courante, sans la modifier.

10.26 GMP

[\[Notes en ligne\]](#)

Ces fonctions vous permettent de travailler avec des nombres de taille arbitraire, en utilisant la librairie GNU **MP**. Pour pouvoir y accéder, vous devez compiler PHP avec le support **GMP** en utilisant l'option `--with-gmp`.

Vous pouvez télécharger **GMP** sur le site de <http://www.swox.com/gmp/>. Ce site propose aussi un manuel **GMP**.

Vous devez utiliser GMP version 2 ou plus récent pour utiliser ces fonctions. Certaines d'entre elles peuvent requérir une version encore plus récente de GMP.

Ces fonctions ont été ajoutée dans PHP 4.0.4.

Note : La plupart des fonctions GMP acceptent des nombres GMP comme arguments, définis ci-dessous comme resource. Cependant, la plus part de ces fonctions acceptent aussi des nombres et des chaînes à partir du moment où on peut les convertir en nombre. Si une fonction utilisant les entiers est plus rapide, elle sera automatiquement appelée si les arguments fournis sont des entiers. Cela se fait de manière transparente : vous pouvez donc utiliser des entiers avec les fonctions GMP sans perte de vitesse. Voir aussi [gmp_init\(\)](#).

Factorielle avec GMP

```
<?php
function fact($x) {
    if($x <= 1)
        return 1;
    else
        return gmp_mul($x, fact($x-1));
}
print gmp_strval(fact(1000))."\n";
?>
```

Cet exemple va calculer factorielle de 1000 (un plutôt grand nombre) très vite.

10.26.1 gmp_init

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_init](#)(mixed *number*)

[gmp_init\(\)](#) crée un nombre GMP, à partir d'un entier ou d'une chaîne. Les chaînes peuvent être en décimal ou en hexadécimal. Dans ce dernier cas, la chaîne doit commencer par 0x.

Création d'un nombre GMP

```
<?php
$a = gmp_init(123456);
$b = gmp_init("0xFFFFDEBACDFEDF7200");
?>
```

Note : Il n'est pas nécessaire d'appeler cette fonction si vous voulez utiliser des entiers ou des chaînes à la place de nombre GMP dans les fonctions GMP, comme par exemple [gmp_add\(\)](#). Les arguments de cette fonction sont automatiquement convertis en nombres GMP, si cette conversion est possible et nécessaire, en utilisant les mêmes règles que [gmp_init\(\)](#).

10.26.2 gmp_intval

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_intval](#)(resource *gmpnumber*)

[gmp_intval\(\)](#) convertit un nombre GMP en entier. [gmp_intval\(\)](#) retourne un résultat cohérent uniquement si le nombre GMP peut être représenté par un entier PHP (i.e. type long signé). Si vous voulez simplement afficher un nombre GMP, utilisez plutôt [gmp_strval\(\)](#).

10.26.3 gmp_strval

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gmp_strval](#)(resource *gmpnumber*, int *base*)

[gmp_strval\(\)](#) convertit un nombre GMP en chaîne de caractères, dans la base *base*. La base par défaut est 10. Les valeurs possibles vont de 2 à 36.

Conversion d'un nombre GMP en chaîne

```
<?php
    $a = gmp_init("0x41682179fbf5");
    printf("Décimal: %s, base 36: %s", gmp_strval($a), gmp_strval($a,36));
?>
```

10.26.4 gmp_add

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_add](#)(resource *a*, resource *b*)

[gmp_add\(\)](#) additionne les nombres GMP *a* et *b*. Le résultat est un nombre GMP.

10.26.5 gmp_sub

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_sub](#)(resource *a*, resource *b*)

[gmp_sub\(\)](#) soustrait le nombre GMP *b* de *a*. Le résultat est un nombre GMP.

10.26.6 gmp_mul

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_mul](#)(resource *a*, resource *b*)

[gmp_mul\(\)](#) multiplie les nombres GMP *a* et *b*. Le résultat est un nombre GMP.

10.26.7 [gmp_div_q](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_div_q](#)(resource *a*, resource *b*, int *round*)

[gmp_div_q\(\)](#) divise le nombre GMP *b* par *a*. Le résultat est un entier. L'arrondi du résultat est défini par *round*, qui peut prendre l'une des valeurs suivantes :

- **GMP_ROUND_ZERO**: Le résultat est tronqué vers 0.
- **GMP_ROUND_PLUSINF**: Le résultat est tronqué vers +infinity
- **GMP_ROUND_MINUSINF**: Le résultat est tronqué vers -infinity

[gmp_div_q\(\)](#) peut aussi être appelée [gmp_div\(\)](#).

Voir aussi [gmp_div_r\(\)](#), [gmp_div_qr\(\)](#).

10.26.8 [gmp_div_r](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_div_r](#)(resource *n*, resource *d*, int *round*)

[gmp_div_r\(\)](#) le reste de la division entière de *n* par *d*. Le reste a le même signe que *n*, s'il est non nul.

Voir [gmp_div_q\(\)](#) pour une description du paramètre *round*.

Voir aussi [gmp_div_q\(\)](#), [gmp_div_qr\(\)](#).

10.26.9 [gmp_div_qr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gmp_div_qr](#)(resource *n*, resource *d*, int *round*)

[gmp_div_qr\(\)](#) divise *n* par *d* et retourne un tableau, dont le premier élément est [n/d] (le quotient entier de la division) et le second est (n - [n/d] * d) (le reste).

Voir [gmp_div_q\(\)](#) pour une description du paramètre *round*.

Division de nombres GMP

```
<?php
$a = gmp_init("0x41682179fbf5");
$res = gmp_div_qr($a, "0xDEFE75");
printf("Le résultat est: q - %s, r - %s", gmp_strval($res[0]), gmp_strval($res[1]));
?>
```

Voir aussi [gmp_div_q\(\)](#), [gmp_div_r\(\)](#).

10.26.10 [gmp_div](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_divexact](#)(resource *a*, resource *b*)

[`gmp_divexact\(\)`](#) est un alias de [`gmp_div_q\(\)`](#).

10.26.11 [gmp_mod](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_mod`](#)(resource *n*, resource *d*)

[`gmp_mod\(\)`](#) calcule *n* modulo *d*. Le résultat est toujours positif ou nul, car le signe de *d* est ignoré.

10.26.12 [gmp_divexact](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_divexact`](#)(resource *n*, resource *d*)

[`gmp_divexact\(\)`](#) divise *n* par *d*, en utilisant les algorithmes de "division exacte". Cette fonction ne fournit de résultats cohérents que s'il est su par avance que *d* divise *n*.

10.26.13 [gmp_cmp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`gmp_cmp`](#)(resource *a*, resource *b*)

[`gmp_cmp\(\)`](#) retourne une valeur positive si *a* > *b*, zéro si *a* = *b* et négative si *a* < *b*.

10.26.14 [gmp_neg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_neg`](#)(resource *a*)

[`gmp_neg\(\)`](#) retourne l'opposé de [`gmp_neg\(\)`](#)*a* (*-a*).

10.26.15 [gmp_abs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_abs`](#)(resource *a*)

[`gmp_abs\(\)`](#) retourne la valeur absolue de *a*.

10.26.16 [gmp_sign](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`gmp_sign`](#)(resource *a*)

[`gmp_sign\(\)`](#) retourne le signe de *a* : 1 si *a* est positif et -1 s'il est négatif.

10.26.17 gmp_fact

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_fact](#) (int *a*)

[gmp_fact\(\)](#) calcule la factorielle de *a* : $a!$.

10.26.18 gmp_sqrt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_sqrt](#) (resource *a*)

[gmp_sqrt\(\)](#) calcule la racine carrée de *a*.

10.26.19 gmp_sqrtrm

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gmp_sqrtrm](#) (resource *a*)

[gmp_sqrtrm\(\)](#) retourne un tableau dont le premier élément est la racine carrée entière de *a* (voir aussi [gmp_sqrt\(\)](#)), et le second est le reste de (i.e., la différence entre *a* et le carré du premier élément).

10.26.20 gmp_perfect_square

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [gmp_perfect_square](#) (resource *a*)

[gmp_perfect_square\(\)](#) retourne TRUE si *a* est un carré parfait, et FALSE sinon.
Voir aussi : [gmp_sqrt\(\)](#), [gmp_sqrtrm\(\)](#).

10.26.21 gmp_pow

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_pow](#) (resource *base*, int *exp*)

[gmp_pow\(\)](#) élève *base* à la puissance *exp*. Dans le cas de 0^0 , [gmp_pow\(\)](#) retourne 1. *exp* ne doit pas être négatif.

10.26.22 gmp_powm

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_powm](#) (resource *base*, resource *exp*, resource *mod*)

[gmp_powm\(\)](#) calcule (*base* puissance *exp*) modulo *mod*. Si *exp* est négatif, le résultat est indéfini.

[10.26.23 gmp_prob_prime](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_prob_prime](#) (resource *a*, int *reps*)

Si [gmp_prob_prime\(\)](#) retourne 0, *a* est défini comme non premier. Si [gmp_prob_prime\(\)](#) retourne 1, alors *a* est "probablement" premier. Si [gmp_prob_prime\(\)](#) retourne 2, alors *a* est sûrement premier. *reps* peut raisonnablement varier de 5 à 10 (par défaut, c'est 10); une valeur supérieure réduit la probabilité qu'un nombre non premier soit identifié comme "probablement" premier. Cette fonction utilise le teste de probabilité Miller–Rabin.

[10.26.24 gmp_gcd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_gcd](#) (resource *a*, resource *b*)

[gmp_gcd\(\)](#) calcule de PGCD (plus grand commun diviseur) de *a* et *b*. Le résultat est toujours positif, même si l'un des deux (ou les deux) nombres est négatif.

[10.26.25 gmp_gcdext](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gmp_gcdext](#) (resource *a*, resource *b*)

[gmp_gcdext\(\)](#) calcule les entiers *g*, *s*, et *t*, tels que $a*s + b*t = g = \text{gcd}(a,b)$, où gcd est le pgcd de *a* et *b*. La fonction retourne un tableau avec les éléments *g*, *s* et *t*.

[10.26.26 gmp_invert](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [gmp_invert](#) (resource *a*, resource *b*)

[gmp_invert\(\)](#) calcule l'invers de *a* modulo *b*. Retourne FALSE si un tel inverse n'existe pas.

[10.26.27 gmp_legendre](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_legendre](#) (resource *a*, resource *p*)

[gmp_legendre\(\)](#) calcule le [symbole de Legendre](#) de *a* et *p*. *p* doit être positif et impair.

[10.26.28 gmp_jacobi](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_jacobi](#) (resource *a*, resource *p*)

[`gmp_jacobi\(\)`](#) calcule le [symbole de Jacobi](#) de *a* et *p*. *p* doit être positif et impair.

[10.26.29 gmp_random](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_random`](#)(int *limiter*)

[`gmp_random\(\)`](#) génère un nombre aléatoire. Ce nombre est limité par *limiter*. Si *limiter* est négatif, un nombre négatif est généré.

[10.26.30 gmp_and](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_and`](#)(resource *a*, resource *b*)

[`gmp_and\(\)`](#) calcule le ET logique de *a* et *b*.

[10.26.31 gmp_or](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_or`](#)(resource *a*, resource *b*)

[`gmp_or\(\)`](#) calcule le OU logique de *a* et *b*.

[10.26.32 gmp_xor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_xor`](#)(resource *a*, resource *b*)

[`gmp_or\(\)`](#) calcule le OU exclusif logique de *a* et *b*.

[10.26.33 gmp_setbit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_setbit`](#)(resource *&a*, int *index*, bool *set_clear*)

[`gmp_setbit\(\)`](#) modifie le bit *index* dans *a*. *set_clear* indique si le bit est mis à 0 ou 1. Par défaut, il est mis à 1.

[10.26.34 gmp_clrbit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`gmp_clrbit`](#)(resource *&a*, int *index*)

[`gmp_clrbit\(\)`](#) met le bit *index* à 0 dans le nombre GMP *a*.

10.26.35 gmp_scan0

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_scan0](#)(resource *a*, int *start*)

[gmp_scan0\(\)](#) recherche dans *a*, en commençant à la position *start*, vers les bits de poids lourd, jusqu'à ce qu'elle rencontre un bit à 0. Sa position est alors retournée.

10.26.36 gmp_scan1

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_scan1](#)(resource *a*, int *start*)

[gmp_scan1\(\)](#) recherche dans *a*, en commençant à la position *start*, vers les bits de poids lourd, jusqu'à ce qu'elle rencontre un bit à 1. Sa position est alors retournée.

10.26.37 gmp_popcount

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_popcount](#)(resource *a*)

[gmp_popcount\(\)](#) dénombre la population de *a*.

10.26.38 gmp_hamdist

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gmp_hamdist](#)(resource *a*, resource *b*)

[gmp_hamdist\(\)](#) retourne la distance de Hamming entre *a* et *a*. Les deux paramètres doivent être strictement positif.

10.27 HTTP

[\[Notes en ligne\]](#)

Ces fonctions permettent de travailler sur les informations transmises au navigateur, via le protocole HTTP.

10.27.1 header

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [header](#)(string *string*)

[PHP 3, PHP 4]

[header\(\)](#) permet de spécifier un entête HTTP lors de l'envoi des fichiers HTML. Reportez vous à [HTTP 1.1 Specification](#) pour plus d'informations sur les entêtes HTTP. NB : la fonction [header\(\)](#) doit être appelée avant la première balise HTML, et avant n'importe quel envoi de commande PHP. C'est une erreur très courante que de lire du code avec la fonction [include\(\)](#) ou avec `auto_prepend` et d'avoir des espaces ou des lignes vides

dans ce code qui produisent un début de sortie avant que [header\(\)](#) n'ait été appelé.

Il y a cependant deux entêtes spéciaux. Le premier est "Location". Non seulement il renvoie un entête au client, mais en plus, il envoie un statut de redirection à Apache. Du point de vue de l'auteur de script, cela importe peu, mais pour ceux qui connaissent les rouages internes d'Apache, c'est primordial.

```
<?php
header("Location: http://www.php.net"); /* Redirige le client vers le site PHP */
exit; /* Assure que le code ci dessous n'est jamais exécuté. */
?>
```

Le deuxième type d'appel spécial regroupe tous les entêtes qui commencent par "HTTP/" (la casse n'est pas importante). Par exemple, si vous avez votre page d'erreur 404 Apache qui pointent sur un script PHP, c'est une bonne idée que de vous assurez que le script PHP génère une erreur 404. La première chose à dans votre script est :

```
header("http/1.0 404 Not Found");
```

Les scripts PHP génèrent souvent du HTML dynamiquement, qui ne doit pas être mis en cache, ni par le client, ni par les proxy intermédiaires. On peut forcer la désactivation du cache de nombreux clients et proxy avec

```
<?php
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // Date du passé
header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT"); // toujours modifié
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Pragma: no-cache"); // HTTP/1.0
?>
```

10.27.2 headers_sent

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [headers_sent](#)(*void*)

[PHP 3 >= 3.0.8, PHP 4 >= 4.0b2]

[headers_sent\(\)](#) retourne TRUE si les entêtes HTTP ont déjà été envoyés, et FALSE sinon.

Voir aussi [header\(\)](#).

10.27.3 setcookie

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [setcookie](#)(string *name*, string *value*, int *expire*, string *path*, string *domain*, int *secure*)

[PHP 3, PHP 4]

[setcookie\(\)](#) définit un cookie qui sera envoyé avec le reste des entêtes. Les cookies doivent passer avant tout autre entête (c'est une restriction des cookies, pas de PHP). Cela vous impose d'appeler cette fonction avant toute balise <html> ou <head>.

Tous les arguments sauf **name** (nom) sont optionnels. Si seul le nom est présent, le cookie portant ce nom sera supprimé du client. Vous pouvez aussi utiliser une chaîne vide comme valeur, pour ignorer un argument. Le paramètre **expire** est un timestamp UNIX, du même genre que celui retourné par [time\(\)](#) ou [mktime\(\)](#). Le paramètre **secure** indique que le cookie doit être uniquement transmis à travers une connexion HTTPS sécurisée.

Erreurs communes:

Les cookies ne seront accessibles qu'au chargement de la prochaine page.

Les appels multiples à [setcookie\(\)](#) dans la même page seront réalisés dans l'ordre inverse. Si vous essayez d'effacer un cookie avant d'insérer une autre valeur, il faut placer l'insertion avant l'effacement.

Quelques exemples :

Exemples avec setcookie()

```
<?php
setcookie("TestCookie","Test Value");
setcookie("TestCookie",$value,time()+3600); /* expire dans une heure */
setcookie("TestCookie",$value,time()+3600,"/~rasmus/", ".utoronto.ca",1);
?>
```

Notez que la partie "valeur" du cookie sera automatiquement encodée URL lorsque vous envoyez le cookie, et lorsque vous le recevez, il sera automatiquement décodé, et affecté à la variable du même nom que le cookie. Pour voir le résultat, essayez les scripts suivants :

```
<?php
echo $TestCookie;
echo $HTTP_COOKIE_VARS["TestCookie"];
?>
```

Vous pouvez aussi utiliser les cookies avec des tableaux, en utilisant la notation des tableaux. Cela a pour effet de créer autant de cookie que votre tableau a d'éléments, mais lorsque les cookies seront reçus par PHP, les valeurs seront placées dans un tableau :

```
setcookie( "cookie[three]", "cookiethree" );
setcookie( "cookie[two]", "cookietwo" );
setcookie( "cookie[one]", "cookieone" );
if ( isset( $cookie ) ) {
while( list( $name, $value ) = each( $cookie ) ) {
echo "$name == $value<br>\n";
}
}
```

Pour d'autres informations sur les cookies, jetez un oeil sur

http://www.netscape.com/newsref/std/cookie_spec.html.

Microsoft Internet Explorer 4 utilisé avec le Service Pack 1 ne gère pas bien les cookies qui possèdent un paramètre **path**.

Netscape Communicator 4.05 et Microsoft Internet Explorer 3.x semblent ne pas gérer correctement les cookies lorsque **path** et **time** ne sont pas fournis.

10.28 Hyperwave

[\[Notes en ligne\]](#)

10.28.1 Introduction

[Notes en ligne]

Hyperwave a été développé par [IICM](#) à Graz. Son nom original était *Hyper-G* et il a pris le nom de Hyperwave lors de sa commercialisation (en 1996, si mes souvenirs sont bons).

Hyperwave n'est pas gratuit. La version actuelle est la 4.1, disponible à www.hyperwave.com. Une version limitée à 30 jours peut être demandée.

HIS est un système d'information similaire à une base de données, (HIS, Hyperwave Information Server). HIS se concentre sur l'enregistrement et la gestion des documents. Un document peut être n'importe quelle donnée, qui peut être stockée dans un fichier. Chaque document est accompagné par un enregistrement. Cet enregistrement contient des méta données à propos du document. Ces méta données sont des listes d'attributs qui peuvent être étendues par l'utilisateur. Un attribut est une paire clé/valeur de la forme : clé =valeur. L'enregistrement complet contient autant de paire que le désire l'utilisateur. Le nom d'un attribut n'a pas besoin d'être unique, c'est à dire qu'une même clé peut apparaître plusieurs fois dans un enregistrement. Cela peut être utile si vous devez donner un titre à votre document en plusieurs langues, par exemple. Dans un cas pareil, la convention est que chaque valeur de titre est précédée par deux lettres et deux points, tel que : 'fr:Titre en français' ou 'ge:Titel in deutsch'. D'autres attributs comme une description ou des mots clés sont aussi susceptibles de recourir à ce genre de procédé. Vous pouvez aussi remplacer l'abréviation de langage par une autre chaîne, tant qu'elle est séparée de la valeur par les deux points.

Chaque enregistrement a une représentation native qui contient toutes les paires clé/valeur, séparées par un retour à la ligne. L'extension Hyperwave reconnaît une autre représentation qui est un tableau associatif, où les attributs servent de clés. Les attributs multilingues étant géré sous la forme d'un autre tableau associatif, dont les clés sont les chaînes de langue. En fait, tous les attributs multiformes sont gérés sous la forme de tableau associatif. (Cela n'est pas encore complètement codé. Uniquement les attributs de titre, description et mot clés sont traités correctement).

En dehors des documents, tous les hyper liens contenus dans un documents sont enregistrés dans un autre enregistrement. Les hyperliens qui sont à l'intérieur d'un document en seront supprimés, et enregistrés dans des objets particuliers, au moment de l'insertion dans la base de données. L'enregistrement des hyper-liens contient les informations d'origine et d'objectif. Afin d'accéder au document original, vous devrez lire le document sans les liens, puis lire les liens, et les réinsérer (les [hw_pipedocument\(\)](#) et [hw_gettext\(\)](#) le font pour vous. L'avantage de séparer les liens du document est évident : une fois qu'un document, cible d'un hyperlien, a été renommé, le liens peut facilement être modifié. Le document contenant le lien n'est pas modifié pour autant. Vous pouvez même ajouter un lien à un document sans le modifier.

Dire que [hw_pipedocument\(\)](#) et [hw_gettext\(\)](#) font l'insertion automatiquement n'est pas aussi simple qu'il n'y paraît. L'insertion implique une certaine hiérarchie de document. Sur un serveur web, la hiérarchie est fournie par le système de fichiers, mais Hyperwave dispose de sa propre hiérarchie et les noms de fichiers ne reflètent pas la position d'un objet dans cette hiérarchie. Ainsi, la création de liens requière en premier lieu la construction de la hiérarchie et de l'espace des noms dans une hiérarchie web et un espace de nom web. La différence fondamentale entre Hyperwave et le web est qu'il y a une distinction claire en les noms et la hiérarchie dans Hyperwave. Le nom ne contient aucune information à propos de la position de l'objet dans la hiérarchie. Sur le web, le nom contient les informations de localisation dans la hiérarchie. Cela conduit à deux méthodes de d'accès : soit la hiérarchie Hyperwave et le nom de l'objet sont inscrit dans l'URL. Pour simplifier les choses, une deuxième approche est pratiquée. L'objet Hyperwave de nom 'mon_objet' correspond à l'URL 'http://hote/mon_objet', peu importe alors où il est rangé dans la hiérarchie. Un objet dont le nom est 'parent/mon_objet' peut être le fils de l'objet 'mon_objet' dans la hiérarchie Hyperwave, bien que ce soit le contraire en convention web, et cela risque de perturber l'utilisateur.

Ayant pris cette décision, un deuxième problème surgit : comment faire l'interface avec PHP ? L' URL http://hote/mon_objet n'appellera aucun script PHP à moins que vous ne demandiez à votre serveur web de le remplacer par autre chose, comme par exemple : 'http://host/php3_script/mon_objet' et le script 'php3_script' utilise la variable \$PATH_INFO pour rechercher l'objet 'mon_objet' sur le serveur Hyperwave. Il y a juste un

petit inconvénient, qui peut facilement être corrigé. Réécrire une URL ne vous permettra aucun accès aux autres documents du serveur web. Un script de recherche dans le serveur Hyperwave serait impossible. Il vous faudra donc au moins une autre règle pour exclure certaines URL, comme par exemple celles qui commencent par `http://host/Hyperwave`. Voici de manière simple, la manière de partager un espace de nom entre un serveur web et un serveur Hyperwave serveur.

Basé sur le mécanisme précédent, on trouve l'insertion dans les documents.

Il est plus compliqué d'avoir PHP ne fonctionne pas comme un module de serveur, ou un script CGI, mais comme une application indépendante. Dans ce cas, il est utile d'inscrire la hiérarchie et le nom de fichier Hyperwave dans le système de fichier. Mais comme cela risque d'entrer en conflit avec le séparateur de dossier ('/'), il faut le remplacer par un autre caractère, '_ '.

Le protocole réseau pour communiquer avec un serveur Hyperwave est appelé [HG-CSP](#) (Hyper-G Client/Server Protocol). Il est basé sur des messages qui initie des actions, comme par exemple, lire l'en-tête de fichier. Dans les premières versions de Hyperwave Server deux clients natifs (Harmony, Amadeus) étaient fournis pour permettre la communication avec le serveur. Ils ont disparu lors de la commercialisation de Hyperwave. En tant qu'ersatz, un client appelé wavemaster est désormais fourni. wavemaster est un espèce de convertisseur de protocole de HTTP en HG-CSP. L'idée est de faire toute l'administration de la base et la visualisation des documents par une interface web. Le wavemaster implémente un jeu d'emplacement pour certaines actions de personnalisation de l'interface. Ce jeu est appelé PLACE language. PLACE pêche encore par le manque de nombreuses fonctions de programmations, et le manque d'évolutivité. Cela a conduit à l'utilisation de JavaScript ce qui ne rend pas la vie facile.

Que PHP supporte Hyperwave permet de combler ces manques. PHP implémente tous les messages définis par HG-CSP mais fournit d'autres commandes puissantes, comme par exemple, celle de lire des documents complets.

Hyperwave dispose de sa propre terminologie pour localiser certaines informations. Cette terminologie a été largement étendue. Presque toutes les fonctions utilisent l'un des types de données suivants :

- object ID: un entier, unique pour chaque objet sur le serveur Hyperwave. C'est aussi un des attributs de l'enregistrement de l'objet (ObjectID). Les object ids sont souvent utilisées comme paramètre d'entrée pour spécifier un objet.
- object record: Une chaîne contenant des paires clé=valeur. Les paires sont séparées par un retour à la ligne. Un enregistrement d'objet peut facilement être converti en tableau, avec la fonction [hw_objrec2array\(\)](#). De nombreuses fonctions retournent un objet records. Ces fonctions ont leur nom qui finit par obj.
- object array: Un tableau associatif qui contient tous les attributs d'un objet. La clé est l'attribut. Si un attribut apparaît plusieurs fois, il sera représenté sous la forme d'un tableau associatif ou indexé. Les attributs qui dépendent des langues (comme title, keyword ou description) seront représentés sous la forme d'un tableau associatif, dont les clés seront les abréviations de langues. Tous les autres attributs à valeur multiple prendront la forme d'un tableau indexé.
- hw_document: Ce type est un nouveau type de données, qui contient le document lui-même, comme par exemple HTML, PDF etc. Il est optimisé pour les documents HTML mais peut s'utiliser avec n'importe quel format.

De nombreuses fonctions qui retournent un tableau d'enregistrements, retournent aussi un tableau associé, avec des informations statistiques. Ce tableau est le dernier élément du tableau d'enregistrements. Les statistiques contiennent les entrées suivantes :

Hidden

- Nombre d'objets dont l'attribut PresentationHints est Hidden.
- CollectionHead

- Nombre d'objet dont l'attribut PresentationHints est CollectionHead.
FullCollectionHead
- Nombre d'objet dont l'attribut PresentationHints est FullCollectionHead.
CollectionHeadNr
- Index du premier objet du tableau d'enregistrement avec l'attribut PresentationHints à CollectionHead.
FullCollectionHeadNr
- Index du premier objet du tableau d'enregistrement avec l'attribut PresentationHints est FullCollectionHead.
Total
- Total: Nombre d'enregistrements.

10.28.2 Intégration avec Apache

[\[Notes en ligne\]](#)

L'extension Hyperwave est utilisée de manière optimale lorsque PHP est compilé comme module Apache. Dans ce cas, le serveur Hyperwave sous jacent peut être caché quasiment totalement aux utilisateurs, si Apache utilise son moteur d'écriture. Les explications suivantes vous éclaireront :

Etant donné que PHP avec l'extension Hyperwave et Apache tend à remplacer la solution native basé sur Wavemaster, je vais supposer que le serveur Apache servira seulement d'interface Hyperwave. Ce n'est pas nécessaire, mais cela simplifie grandement la configuration. Le concept est très simple. Premièrement, vous avez besoin d'un script PHP qui évalue la variable **PATH_INFO** et considère que cette valeur est un objet Hyperwave. Appelons ce script 'Hyperwave'. L'URL

`http://votre.hote/Hyperwave/nom_objet` retourne alors l'objet Hyperwave dont le nom est 'nom_objet'. Le script doit alors réagir suivant le type de l'objet. Si c'est un groupe, il devra probablement retourner une liste de fils. Si c'est un document, il pourra retourner son type MIME et son contenu. Une amélioration peut être obtenue en utilisant le moteur de réécriture d'Apache. D'un point de vue utilisateur, il est plus direct si l'URL `http://votre.hote/nom_objet` retourne l'objet. La règle de réécriture est simple :

```
RewriteRule ^/(.*) /usr/local/apache/htdocs/HyperWave/$1 [L]
```

Maintenant toutes les URL pointent sur un objet Hyperwave. Cela conduit à un problème simple. Il n'y a pas d'autre façon d'exécuter, c'est à dire rechercher, un autre script que ce script 'Hyperwave'. Cela pourra être corrigé avec une autre règle telle que:

```
RewriteRule ^/hw/(.*) /usr/local/apache/htdocs/hw/$1 [L]
```

Le dossier `/usr/local/apache/htdocs/hw` sera ainsi réservé pour d'autres scripts et fichiers. Assurez vous que cette règle est évaluée avant la première règle que nous avons définie. Il y a juste un léger inconvénient : tous les objets Hyperwave qui commencent par 'hw/' seront cachés. Alors, assurez vous que vous n'utilisez pas de tels noms. Si vous avez besoin d'autres dossiers, par exemple, un dossier d'images, ajoutez simplement d'autres règles. N'oubliez pas de lancer le moteur de réécriture avec

```
RewriteEngine on
```

Mon expérience m'a montré que vous aurez besoin des scripts suivants :

- Retourne l'objet lui même

- Pour autoriser la recherche
- S'identifier
- Choisir une configuration
- Un script pour chaque fonction supplémentaire, comme afficher un objet, afficher des informations sur les utilisateurs, afficher le statut du serveur, etc...

10.28.3 A faire

[\[Notes en ligne\]](#)

Il reste encore beaucoup à faire :

- La fonction `hw_InsertDocument` doit être séparée en deux : [hw_insertobject\(\)](#) et `@xref{function.hw-putdocument, , hw_putdocument()}`.
- Les noms de nombreuses fonctions ne sont pas encore fixés.
- La plupart des fonctions requièrent la connexion courante comme premier paramètre. Cela conduit à beaucoup de frappe clavier, même si il n'y a souvent qu'une seule connexion en jeu. Une connexion par défaut améliorerait ceci.

10.28.4 hw_array2objrec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_array2objrec](#)(array *object_array*)
[PHP 3>= 3.0.4, PHP 4]

[hw_array2objrec\(\)](#) convertit un *object_array* en un objet. Les attributs multiples tels que 'Title' en différentes langues seront traités correctement.
Voir aussi [hw_objrec2array\(\)](#).

10.28.5 hw_children

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_children](#)(int *connection*, int *objectID*)
[PHP 3>= 3.0.3, PHP 4]

[hw_children\(\)](#) retourne un tableau avec des object ids. Chaque object id est celui d'un des fils du groupe dont l'id est *objectID*. Ce tableau contient tous les fils, documents et groupes.

10.28.6 hw_childrenobj

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_childrenobj](#)(int *connection*, int *objectID*)
[PHP 3>= 3.0.3, PHP 4]

[hw_childrenobj\(\)](#) retourne un tableau avec des object records. Chaque object records est celui d'un des fils du

groupe dont l'id est *objectID*. Ce tableau contient tous les fils, documents et groupes

10.28.7 hw_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_close](#) (int *connection*)

[PHP 3>= 3.0.3, PHP 4]

[hw_close\(\)](#) retourne FALSE si la connexion n'est pas valide, et sinon, TRUE. Ferme la connexion *connection* à un serveur Hyperwave.

10.28.8 hw_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_connect](#) (string *host*, int *port*, string *username*, string *password*)

[PHP 3>= 3.0.3, PHP 4]

[hw_connect\(\)](#) ouvre une connexion Hyperwave et retourne un identifiant de connexion, en cas de succès, ou FALSE, si la connexion n'a pas pu être créée. Chaque argument doit être entouré de guillemets, sauf le numéro de port. Les arguments *username* et *password* sont optionnels, et peuvent être ignorés. Dans ce cas, aucune identification ne sera faite au niveau du serveur. Cela revient à s'identifier en tant qu'utilisateur anonyme. Cette fonction retourne un identifiant de connexion qui sera nécessaire aux autres fonctions Hyperwave. Vous pouvez avoir plusieurs connexions simultanées. N'oubliez pas que les mots de passe ne sont pas cryptés.

Voir aussi [hw_pconnect\(\)](#).

10.28.9 hw_cp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_cp](#) (int *connection*, array *object_id_array*, int *destination_id*)

[PHP 3>= 3.0.3, PHP 4]

[hw_cp\(\)](#) copie les objet ayant les objects id *object_id_array*, et crée un groupe ayant l'object id *destination_id*.

La valeur retournée est le nombre d'objets copiés.

Voir aussi [hw_mv\(\)](#).

10.28.10 hw_deleteobject

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_deleteobject](#) (int *connection*, int *object_to_delete*)

[PHP 3>= 3.0.3, PHP 4]

[hw_deleteobject\(\)](#) efface l'objet dont l'identifiant est *object_to_delete*. Toutes les instances de l'objets seront effacées.

Retourne TRUE si aucune erreur ne survient, et sinon, FALSE.

Voir aussi [hw_mv\(\)](#).

[10.28.11 hw_docbyanchor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_docbyanchor](#) (int *connection*, int *anchorID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_docbyanchor\(\)](#) retourne l'identifiant d'objet de l'objet dans l'ancrage *anchorID*.

[10.28.12 hw_docbyanchorobj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_docbyanchorobj](#) (int *connection*, int *anchorID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_docbyanchorobj\(\)](#) retourne les attributs du document qui correspond à *anchorID*.

[10.28.13 hw_documentattributes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentattributes](#) (int *hw_document*)

[PHP 3>= 3.0.3, PHP 4 <= 4.0b1]

[hw_documentattributes\(\)](#) retourne les attributs du document.

Voir aussi [hw_documentbodytag\(\)](#), [hw_documentsize\(\)](#).

[10.28.14 hw_documentbodytag](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentbodytag](#) (int *hw_document*)

[PHP 3>= 3.0.3, PHP 4 <= 4.0b1]

[hw_documentbodytag\(\)](#) retourne la balise BODY du document. Si le document est un document HTML , la balise BODY doit être affichée avant le document.

Voir aussi [hw_documentattributes\(\)](#), [hw_documentsize\(\)](#).

[10.28.15 hw_documentcontent](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentcontent](#) (int *hw_document*)

[PHP 3>= 3.0.8]

[hw_documentcontent\(\)](#) retourne la balise BODY du document. Si le document est un document HTML , la balise BODY doit être affichée avant le document.

Voir aussi [hw_documentattributes\(\)](#), [hw_documentsize\(\)](#), [hw_documentsetcontent\(\)](#).

10.28.16 hw_documentsetcontent

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_documentsetcontent](#) (int *hw_document*, string *content*)

[PHP 3>= 3.0.8]

[hw_documentsetcontent\(\)](#) modifie/remplace le contenu d'un document. Si le document est un document HTML, le contenu représente tout qui est placé au delà de la balise BODY. Les informations de HEAD et de la balise BODY sont enregistrées dans les attributs. Si vous fournissez aussi ces informations dans le corps du document, le serveur Hyperwave modifiera les attributs. Cela n'est cependant pas une bonne idée. Si la fonction échoue, l'ancien contenu est restauré.

Voir aussi [hw_documentattributes\(\)](#), [hw_documentsize\(\)](#), [hw_documentcontent\(\)](#).

10.28.17 hw_documentsize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_documentsize](#) (int *hw_document*)

[PHP 3>= 3.0.3, PHP 4 <= 4.0b1]

[hw_documentsize\(\)](#) retourne la taille du document en octets.

Voir aussi [hw_documentbodytag\(\)](#), [hw_documentattributes\(\)](#).

10.28.18 hw_errormsg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_errormsg](#) (int *connection*)

[PHP 3>= 3.0.3, PHP 4]

[hw_errormsg\(\)](#) retourne une chaîne contenant le dernier message d'erreur, ou 'No Error' (pas d'erreur). Si FALSE est retourné, cette fonction a échoué. Ce message est relatif à la dernière commande exécutée.

10.28.19 hw_edittest

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_edittest](#) (int *connection*, int *hw_document*)

[PHP 3>= 3.0.3, PHP 4]

[hw_edittest\(\)](#) charge le texte du document sur le serveur. Les attributs du document ne doivent pas être modifiés tant que le document est en train d'être édité. Cette fonction n'est disponible que sur les documents texte. Elle n'ouvrira pas de canal de transfert, et donc, bloquera le script durant le transfert.

Voir aussi [hw_pipedocument\(\)](#), [hw_free_document\(\)](#), [hw_documentbodytag\(\)](#), [hw_documentsize\(\)](#), [hw_outputdocument\(\)](#), [hw_gettext\(\)](#).

10.28.20 hw_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_error](#) (int *connection*)

[PHP 3>= 3.0.3, PHP 4]

[hw_error\(\)](#) retourne le code d'erreur de la dernière erreur. Si la valeur 0 est retournée, c'est qu'il n'y avait pas d'erreur. L'erreur se rapporte à la dernière commande.

10.28.21 hw_free_document

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_free_document](#)(int *hw_document*)

[PHP 3>= 3.0.3, PHP 4]

[hw_free_document\(\)](#) détruit un document Hyperwave.

10.28.22 hw_getparents

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getparentsobj](#)(int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getparentsobj\(\)](#) retourne un tableau indexé avec les identifiants des objets parents de *objectID*.

10.28.23 hw_getparentsobj

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getparentsobj](#)(int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getparentsobj\(\)](#) retourne un tableau indexé, avec les attributs et un tableau associé, d'informations statistiques à propos des attributs. Ce tableau associé est le dernier élément du tableau retourné. Chaque attribut appartient au père de l'objet *objectID*.

10.28.24 hw_getchildcoll

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchildcoll](#)(int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getchildcoll\(\)](#) retourne un tableau contenant les identifiants d'objets des groupes fils du groupe *objectID*. Cette fonction ne retournera pas d'identifiants d'objets des documents fils. Voir aussi [hw_getchilddoccoll\(\)](#).

10.28.25 hw_getchildcollobj

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchildcollobj](#)(int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getchildcollobj\(\)](#) retourne un tableau d'object record. Chaque object records appartient à un groupe d'enfants de la collection *objectID*. La fonction ne retournera pas de documents enfants.

Voir aussi [hw_childrenobj\(\)](#), [hw_getchilddoccollobj\(\)](#).

10.28.26 [hw_getremote](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_getremote](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getremote\(\)](#) retourne un document distant. Les documents distants sont , en Hyperwave, des documents lus depuis une source externe. La plus part des documents éloignés sont des pages web externes, ou des requêtes sur une base de données. Afin de pouvoir accéder à des sources externes, grâce aux documents distants, Hyperwave introduit l'interface HGI (Hyperwave Gateway Interface) qui est similaire à CGI.

Actuellement, seuls les protocoles de FTP, HTTP et certaines bases de données sont accessibles avec HGI.

[hw_getremote\(\)](#) retourne le document de la source distante. Si vous voulez utiliser cette fonction, il vous faut vous familiariser avec HGIs. Il est aussi préférable d'utiliser PHP plutôt que Hyperwave pour accéder aux sources externes. Le support des bases de données sera plus difficile avec Hyperwave que PHP.

Voir aussi [hw_getremotechildren\(\)](#).

10.28.27 [hw_getremotechildren](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_getremotechildren](#) (int *connection*, string *object record*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getremotechildren\(\)](#) retourne les fils d'un document distant. Les fils d'un document distant sont des documents distants eux mêmes. Cela est cohérent si une requête sur une base de données doit être rendu plus sélective, comme expliqué dans Hyperwave Programmers' Guide. Si le nombre de fils est 1 la fonction va retourner le document lui même, la fonction retournera le document lui même, formaté Hyperwave Gateway Interface (HGI). Si le nombre de fils est supérieur 1 la fonction retournera un tableau d'attributs, qui pourra servir à une nouvelle requête avec [hw_getremotechildren\(\)](#). Ces attributs sont virtuels et n'existent pas sur le serveur Hyperwave, et ainsi, n'ont pas d'identifiant d'objet valide. L'ordre exact de ces objets est du ressort de HGI. Si vous voulez utiliser cette fonction, vous devez être très familier HGIs. Il vaut mieux PHP plutôt que Hyperwave pour accéder aux fichiers distants. Le support de base de données y est bien meilleur.

Voir aussi [hw_getremote\(\)](#).

10.28.28 [hw_getsrcbydestobj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getsrcbydestobj](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getsrcbydestobj\(\)](#) retourne les attributs de tous les ancrages qui pointent sur *objectID*. L'objet peut être un document ou un autre ancrage, de type destination.

Voir aussi [hw_getanchors\(\)](#).

[10.28.29 hw_getobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobject](#)(int *connection*, [int]array] *objectID*, string *query*)
[PHP 3>= 3.0.3, PHP 4]

[hw_getobject\(\)](#) retourne les attributs de l'objet dont l'identifiant est *objectID*, si le second paramètre est un entier. Si le second paramètre est un tableau, la fonction retournera un tableau d'attributs. Dans ce cas, le dernier paramètre est aussi évalué.

query a la syntaxe suivante :

```
<expression> ::= "(" <expression> ")" |
"!" <expression> | /* NOT */
<expression> "||" <expression> | /* OR */
<expression> "&&" <expression> | /* AND */
<attribute> <operator> <value>
<attribute> ::= /* * n'importe quel attribut (Title, Author, DocumentType ...) */
<operator> ::= "=" | /* égal */
"<" | /* moins que (comparaison de type chaîne) */
">" | /* plus que (comparaison de type chaîne) */
"~" /* recherche par expression régulière */
```

query permet de sélectionner une nouvelle fois certains objets dans la liste des objets donnés. Contrairement aux autres requêtes, celle ci peut utiliser des attributs non indexés. Le nombre d'attributs retourné dépend de la requête de la requête, et des autorisations d'accès aux objets.

Voir aussi [hw_getandlock\(\)](#), [hw_getobjectbyquery\(\)](#).

[10.28.30 hw_getandlock](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_getandlock](#)(int *connection*, int *objectID*)
[PHP 3>= 3.0.3, PHP 4]

[hw_getandlock\(\)](#) retourne les attributs, et verrouille l'objet *objectID*. Le verrouillage empêchera les autres utilisateurs d'y accéder, jusqu'à ce qu'il soit deverrouillé.

Voir aussi [hw_unlock\(\)](#), [hw_getobject\(\)](#).

[10.28.31 hw_gettext](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_gettext](#)(int *connection*, int *objectID*, mixed *rootID/prefix*)
[PHP 3>= 3.0.3, PHP 4]

[hw_gettext\(\)](#) retourne le document de l'objet *objectID*. Si le document possède des ancrages qui peuvent être insérés, ils seront déjà insérés. L'option *rootID/prefix* peut être une chaîne ou un entier. Si c'est un entier, il détermine la méthode d'insertion des liens dans le document. Par défaut, il vaut 0 et les liens seront construits en fonction du nom de l'objet cible. Cela sert beaucoup dans les applications web. Si un lien pointe sur un objet avec le nom 'film_internet' le lien HTML sera . La position réelle de la source et de la cible dans la hiérarchie seront ignorés. Vous devrez modifier votre site web pour qu'il

réécrite les URL, comme par exemple '/mon_script.php3/film_internet'. 'mon_script.php3' devra analyser \$PATH_INFO et savoir rechercher le document '/mon_script.php3/film_internet'. Si vous ne voulez pas de ce comportement, vous pouvez affecter à rootID/prefix n'importe quel préfixe. Dans ce cas, ce sera une chaîne. Si *rootID/prefix* est un entier différent de 0 le lien sera construit avec tous les noms de la hiérarchie, en commençant à l'objet d'identifiant rootID/prefix, et séparé par des slash. Si, par exemple, le document 'film_internet' est situé à 'a-b-c-internet_movie' et '-' qui sert de séparateur hiérarchique de niveau sur le serveur Hyperwave et le document source est situé dans 'a-b-d-source' alors, le lien HTML serait: . Cela est très pratique si vous voulez télécharger tout le contenu d'un serveur sur un disque, et faire une carte du système sur votre disque.

[hw_gettext\(\)](#) n'est opérationnelle qu'avec des documents de pur texte. Elle n'ouvrira pas de canal spécial de transfert, et ainsi, bloquera le script le temps du transfert.

Voir aussi [hw_pipedocument\(\)](#), [hw_free_document\(\)](#), [hw_documentbodytag\(\)](#), [hw_documentsize\(\)](#), [hw_outputdocument\(\)](#).

[10.28.32 hw_getobjectbyquery](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyquery](#) (int *connection*, string *query*, int *max_hits*)
[PHP 3>= 3.0.3, PHP 4]

[hw_getobjectbyquery\(\)](#) recherche un objet sur tout le serveur et retourne un tableau d' object ids. Le nombre maximum d'objet est limité par *max_hits*. Si *max_hits* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyqueryobj\(\)](#).

[10.28.33 hw_getobjectbyqueryobj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyqueryobj](#) (int *connection*, string *query*, int *max_hits*)
[PHP 3>= 3.0.3, PHP 4]

[hw_getobjectbyqueryobj\(\)](#) recherche un objet sur tout le serveur et retourne un tableau d' object records. Le nombre maximum d'objet est limité par *max_hits*. Si *max_hits* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyquery\(\)](#).

[10.28.34 hw_getobjectbyquerycoll](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyquerycoll](#) (int *connection*, int *objectID*, string *query*, int *max_hits*)
[PHP 3>= 3.0.3, PHP 4]

[hw_getobjectbyquerycoll\(\)](#) recherche un objet sur tout le groupe objectID et retourne un tableau d' object records. Le nombre maximum d'objet est limité par *objectID*. Si *objectID* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyquerycollobj\(\)](#).

[10.28.35 hw_getobjectbyquerycollobj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getobjectbyquerycollobj](#) (int *connection*, int *objectID*, string *query*, int *max_hits*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getobjectbyquerycollobj\(\)](#) recherche un objet sur tout le groupe *objectID* et retourne un tableau d' object records. Le nombre maximum d'objet est limité par *objectID*. Si *objectID* vaut -1 il n'y a pas de limite.

La requête ne fonctionnera qu'avec des attributs indexés.

Voir aussi [hw_getobjectbyquerycoll\(\)](#).

[10.28.36 hw_getchilddoccoll](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchilddoccoll](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getchilddoccoll\(\)](#) retourne un tableau avec les id des documents fils d'une collection.

Voir aussi [hw_getchildcoll\(\)](#).

[10.28.37 hw_getchilddoccollobj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getchilddoccollobj](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getchilddoccollobj\(\)](#) retourne un tableau contenant les attributs des documents fils du groupe *objectID*.

Voir aussi [hw_childrenobj\(\)](#), [hw_getchildcollobj\(\)](#).

[10.28.38 hw_getanchors](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getanchors](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getanchors\(\)](#) retourne un tableau contenant les identifiants des ancrages du document *objectID*.

[10.28.39 hw_getanchorsobj](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_getanchorsobj](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getanchorsobj\(\)](#) retourne un tableau d'attributs des ancrages du document *objectID*.

10.28.40 hw_mv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_mv](#) (int *connection*, array *object id array*, int *source id*, int *destination id*)
[PHP 3>= 3.0.3, PHP 4]

[hw_mv\(\)](#) déplace les objets dont les identifiants sont passés dans le tableau *source id*, depuis le *source id* dans le *destination id*. Si destination id vaut 0, les objets ne seront plus insérés dans le groupe (ni dans le serveur). Dans ce cas, si une instance était la dernière instance d'un objet, l'objet sera effacé. Si vous voulez effacer toutes les instances d'un coup, utilisez [hw_deleteobject\(\)](#).

La valeur retournée est le nombre d'objet déplacés.

Voir aussi [hw_cp\(\)](#), [hw_deleteobject\(\)](#).

10.28.41 hw_identify

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_identify](#) (string *username*, string *password*)
[PHP 3>= 3.0.3, PHP 4]

[hw_identify\(\)](#) identifie un utilisateur, dont le nom d'utilisateur est *username* et le mot de passe *password*. L'identification n'est valide que pour la session en cours. Je ne pense pas que cette fonction serve souvent. Dans la plus part des cas, il est plus simple de s'identifier lors de l'ouverture de la connexion.

Voir aussi [hw_connect\(\)](#).

10.28.42 hw_incollections

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_incollections](#) (int *connection*, array *object_id_array*, array *collection_id_array*, int *return_collections*)
[PHP 3>= 3.0.3, PHP 4]

[hw_incollections\(\)](#) vérifie qu'un ensemble d'objets (documents ou groupes) donnés par *object_id_array* fait partie des groupe listés par *object_id_array*. Lorsque le quatrième paramètre *return_collections* vaut 0, le sous ensemble d'identifiants qui font partie d'un groupe (i.e. les documents ou groupes qui sont fils d'un ou plusieurs groupe, ou leurs fils, récursivement) est retourné sous la forme d'un tableau. Cette option permet de mettre en valeur la partie de l'arborescence qui contient le résultat d'une requête, dans un sens graphique.

10.28.43 hw_info

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_info](#) (int *connection*)
[PHP 3>= 3.0.3, PHP 4]

[hw_info\(\)](#) retourne les informations de la connexion courante. La chaîne retournée a le format suivant : <Serverstring>, <Host>, <Port>, <Username>, <Port of Client>, <Byte swapping>

10.28.44 [hw_inscoll](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_inscoll](#) (int *connection*, int *objectID*, array *object_array*)

[PHP 3>= 3.0.3, PHP 4]

[hw_inscoll\(\)](#) insère un nouveau groupe, avec les attributs *object_array* dans le groupe *objectID*.

10.28.45 [hw_insdoc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_insdoc](#) (int *connection*, int *parentID*, string *object_record*, string *text*)

[PHP 3>= 3.0.3, PHP 4]

[hw_insdoc\(\)](#) insère un nouveau document avec les attributs *object_record*, dans le groupe *parentID*. Cette fonction insère soit un objet avec ses seuls attributs, soit pur objet ascii, avec *text* si il est fourni. Si vous voulez insérer un document de type général, utilisez plutôt [hw_insertdocument\(\)](#).

Voir aussi [hw_insertdocument\(\)](#), [hw_inscoll\(\)](#).

10.28.46 [hw_insertdocument](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_insertdocument](#) (int *connection*, int *parent_id*, int *hw_document*)

[PHP 3>= 3.0.3, PHP 4]

[hw_insertdocument\(\)](#) insère un document dans le groupe *parent_id*. Le document doit avoir été créé auparavant avec [hw_new_document\(\)](#). Assurez vous que les attributs du nouveau document contiennent au moins les attributs suivants : Type, DocumentType, Title et Name. Vous aurez aussi parfois besoin de MimeType. La fonction retourne l'identifiant de l'objet inséré, ou bien FALSE.

Voir aussi [hw_pipedocument\(\)](#).

10.28.47 [hw_insertobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_insertobject](#) (int *connection*, string *object_rec*, string *parameter*)

[PHP 3>= 3.0.3, PHP 4]

[hw_insertobject\(\)](#) insère un objet dans le serveur. L'objet peut être n'importe quel objet Hyperwave valide. Reportez vous à la documentation HG-CSP pour plus de détails sur les paramètres.

Note: Si vous voulez insérer un ancre, l'attribut Position doit être mis à la valeur start/end (début ou fin) ou encore 'invisible'. Les positions invisibles sont nécessaire si l'annotation n'a pas de liens correspondant dans le texte de l'annotation.

Voir aussi [hw_pipedocument\(\)](#), [hw_insertdocument\(\)](#), [hw_insdoc\(\)](#), [hw_inscoll\(\)](#).

10.28.48 [hw_mapid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_mapid](#)(int *connection*, int *server id*, int *object id*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[hw_mapid\(\)](#) représente l'id d'un objet global de n'importe quel serveur Hyperwave, même si vous ne vous y êtes pas connecté avec [hw_connect\(\)](#), avec un id d'objet local virtuel. Cet id d'objet local peut alors être utilisé comme n'importe quel id d'objet : par exemple on peut obtenir l'enregistrement d'objet avec la fonction [hw_getobject\(\)](#). L'id du serveur est la première partie de l'id global (GOid) de l'objet, qui est en fait une adresse IP.

Note: Afin d'utiliser cette fonction, vous devez lever le flag F_DISTRIBUTED, ce qui ne peut être fait qu'à la compilation. Par défaut, il n'est pas levé. Lisez les commentaires dans le fichier hg_comm.c

10.28.49 [hw_modifyobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_modifyobject](#)(int *connection*, int *object_to_change*, array *remove*, array *add*, int *mode*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[hw_modifyobject\(\)](#) permet d'effacer, d'ajouter ou de modifier les attributs d'un objet. L'objet est repéré par son identifiant *object_to_change*. Le premier tableau, *remove*, est la liste des attributs à effacer. Le deuxième tableau *add* est la liste des attributs à ajouter. Afin de modifier un attribut, il vous faudra d'abord l'effacer, puis l'ajouter à nouveau. [hw_modifyobject\(\)](#) effacera toujours les attributs avant de les ajouter, à moins que la valeur de l'attribut à effacer ne soit pas une chaîne, ou un tableau.

Le dernier paramètre détermine si la modification est récursive ou pas. 1 signifie que la modification est récursive. Si un objet ne peut pas être modifié, il sera ignoré. [hw_error\(\)](#) n'indiquera alors pas toujours d'erreur, même si certains objets n'ont pas pu être modifiés.

Les clés des deux tableaux sont les noms des attributs. La valeur de chaque élément peut être un tableau, ou une chaîne ou n'importe quoi d'autre. Dans le cas du tableau, la valeur de l'attribut est construite en séparant chaque élément par un point virgule. Dans le cas de la chaîne, elle sert directement de valeur. Une chaîne vide provoquera un effacement de l'attribut. Dans le cas où la valeur n'est ni un tableau, ni une chaîne, aucune opération ne sera effectuée. Cela est nécessaire si vous voulez ajouter un attribut complètement une nouvelle valeur pour un attribut existant. Si le tableau d'effacement contenait une chaîne vide comme attribut, le serveur tenterait d'effacer l'attribut, ce qui échouerait de toute manière, car cet attribut n'existe pas. L'ajout de cet attribut échouerait aussi. Affecter la valeur de 0 à cet attribut ne l'effacerait pas, et l'ajout fonctionnerait. Si vous voulez changer l'attribut 'Nom' de valeur courante 'livres' en 'articles' vous devrez faire deux tableaux, et appeler [hw_modifyobject\(\)](#).

Modification d'un attribut

```
<?php
// $connect est une connexion valide
// $objid est l'identifiant de l'objet
$remarr = array("Name" => "books");
$addarr = array("Name" => "articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

Afin d'effacer/ajouter une paire nom=valeur aux attributs d'un objet, utilisez simplement les tableaux

d'effacement et d'ajout, et laissez le dernier/troisième paramètre vide. Si l'attribut est le premier de ce nom à ajouter, donnez une valeur entière à cet élément.

Ajouter un nouvel attribut

```
<?php
// $connect st une connexion Hyperwave valide
// $objid est l'identifiant de l'objet à modifier
$remarr = array("Name" => 0);
$addarr = array("Name" => "articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

Note : Les attributs multilingues, (tels que 'Title'), peuvent être modifiés de deux manières : soit en fournissant la valeur de ces attributs de manière native (langue :valeur), soit en fournissant un tableau avec les éléments de chaque langue, comme décrit ci-dessus. L'exemple deviendrai alors :

Modifier l'attribut de Titre (Title)

```
<?php
$remarr = array("Title" => "en:Books");
$addarr = array("Title" => "en:Articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

ou

Modifier l'attribut Title

```
<?php
$remarr = array("Title" => array("en" => "Books"));
$addarr = array("Title" => array("en" => "Articles", "ge"=>"Artikel"));
$hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

Pour supprimer l'entrée française 'Livres' et ajouter l'entrée 'Articles' et l'entrée allemande 'Artikel'.

Suppression d'un attribut

```
<?php
$remarr = array("Title" => "");
$addarr = array("Title" => "en:Articles");
$hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

Note : Cet exemple va effacer tous les attributs avec le nom 'Title' et ajouter un nouvel attribut 'Title'. Cela peut être pratique pour effacer des attributs récursivement.

Note : Si vous devez effacer tous les attributs avec un certains nom, vous devez passer une chaîne vide comme valeur.

Note : Seuls les attributs 'Title', 'Description' et 'Keyword' gère correctement le préfixe de langue. Pour les autres attributs qui ne portent pas de préfixe de langage, le préfixe 'xx' sera assigné.

Note : L'attribut 'Name' est un peu particulier. Dans certains cas, il ne peut pas être complètement effacé. Vous aurez alors le message 'Change of base attribute' (l'apparition de cette erreur n'est pas très claire). Ainsi, vous aurez à ajouter une nouvelle entrée pour Name puis, effacer l'ancien.

Note : Il ne faut pas encadrer cette fonction par des appels à [hw_getandlock\(\)](#) et [hw_unlock\(\)](#). [hw_modifyobject\(\)](#) le fait de manière interne.

Retourne TRUE si aucune erreur ne survient, et FALSE sinon.

10.28.50 [hw_new_document](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_new_document](#) (string *object_record*, string *document_data*, int *document_size*)
[PHP 3>= 3.0.3, PHP 4]

[hw_new_document\(\)](#) retourne un nouveau document Hyperwave avec comme données *document_data* et comme attributs *object_record*. La longueur de *document_data* doit être donnée dans *document_size*. Cette fonction n'insère pas l'objet dans le serveur Hyperwave.

Voir aussi [hw_free_document\(\)](#), [hw_documentsize\(\)](#), [hw_documentbodytag\(\)](#), [hw_outputdocument\(\)](#), [hw_insertdocument\(\)](#).

10.28.51 [hw_objrec2array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [hw_objrec2array](#) (string *object_record*)
[PHP 3>= 3.0.3, PHP 4]

[hw_objrec2array\(\)](#) convertit les attributs *object_record* d'un objet en un tableau. Les clés du tableau seront les noms des attributs. Les attributs multiples comme par exemple 'Title', dans différentes langues, seront rassemblées dans un autre tableau. Une clé est la partie gauche d'un attribut. Actuellement, seuls les attributs 'Title', 'Description' et 'Keyword' sont traités correctement.

Voir aussi [hw_array2objrec\(\)](#).

10.28.52 [hw_outputdocument](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_outputdocument](#) (int *hw_document*)
[PHP 3>= 3.0.3, PHP 4 <= 4.0b1]

[hw_outputdocument\(\)](#) affiche *hw_document* sans la balise BODY.

10.28.53 [hw_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_pconnect](#) (string *host*, int *port*, string *username*, string *password*)
[PHP 3>= 3.0.3, PHP 4]

[hw_pconnect\(\)](#) retourne un index de connexion en cas de succès, et FALSE si la connexion n'a pas pu être créée. Ouvre une connexion persistante à un serveur Hyperwave. Tous les arguments doivent être entre guillemets, hormis le numéro de port (port). Les arguments *username* et *password* sont optionnels, et peuvent être ignorés. Dans ce cas, aucune authentification ne sera faite, (connexion anonyme). Cette fonction retourne un index de connexion qui sera utilisée par les autres fonctions Hyperwave. Vous pouvez ouvrir plusieurs connexions simultanées.

Voir aussi [hw_connect\(\)](#).

[10.28.54 hw_pipedocument](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_pipedocument](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_pipedocument\(\)](#) retourne le document Hyperwave d'objet id *objectID*. Si le document a des ancrages, ils seront insérés. Le document sera transmis via une connexion de données spéciale, qui ne bloque pas la connexion de contrôle (ie, le serveur n'attend pas la fin du transfert pour rendre la main).

Voir aussi [hw_gettext\(\)](#) (insertions), [hw_free_document\(\)](#), [hw_documentsize\(\)](#), [hw_documentbodytag\(\)](#), [hw_outputdocument\(\)](#).

[10.28.55 hw_root](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_root](#) ()

[PHP 3>= 3.0.3, PHP 4]

[hw_root\(\)](#) retourne l' object id de la racine. Actuellement, cette identifiant est toujours 0. L'ensemble des fils de la racine est celui du serveur courant.

[10.28.56 hw_unlock](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_unlock](#) (int *connection*, int *objectID*)

[PHP 3>= 3.0.3, PHP 4]

[hw_unlock\(\)](#) déverrouille un document, et laisse l'accès aux autres utilisateurs.

Voir aussi [hw_getandlock\(\)](#).

[10.28.57 hw_who](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hw_who](#) (int *connection*)

[PHP 3>= 3.0.3, PHP 4]

[hw_who\(\)](#) retourne un tableau contenant la liste des utilisateurs actuellement connectés au serveur Hyperwave. Chaque élément du tableau est lui même un tableau, qui contient l'identifiant de l'élément, le nom, le système, la date de connexion (onSinceDate), l'heure de connexion (onSinceTime), la durée de connexion (TotalTime) et self. 'self' contient l'élément est l'utilisateur qui a appelé la fonction.

[10.28.58 hw_username](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hw_getusername](#) (int *connection*)

[PHP 3>= 3.0.3, PHP 4]

[hw_getusername\(\)](#) retourne le nom d'utilisateur utilisé par la connexion.

10.29 Fonctions InterBase

[\[Notes en ligne\]](#)

Interbase est une base de données populaire, créée par Borland/Inprise. Pour plus d'informations sur Interbase, allez à <http://www.interbase.com>. Par ailleurs, Interbase vient de rejoindre le mouvement Open Source!

Note : *Le support intégral de InterBase 6 a été ajouté à PHP 4.0.*

Cette base de données utilise les guillemets simples (') pour échapper les caractères, un peu comme le fait Sybase. Ajoutez à votre fichier ``php.ini'` la directive suivante :

```
magic_quotes_sybase = On
```

10.29.1 ibase_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_connect](#) (string *database*, string *username* , string *password* , string *charset* , int *buffers* , int *dialect* , string *role*)

[PHP 3>= 3.0.6, PHP 4]

[ibase_connect\(\)](#) établit une connexion avec un serveur InterBase. *database* doit être un chemin valide jusqu'à un fichier de base de données sur le serveur sur lequel il réside. Si le serveur est distant, il faut le préfixer avec un nom d'hôte 'hostname:' (TCP/IP), '//hostname/' (NetBEUI) ou 'hostname' (IPX/SPX), en fonction du protocole de communication utilisé. *username* et *password* peuvent être spécifiés dans les directives de configuration du PHP `ibase.default_user` et `ibase.default_password`. *charset* est le jeu de caractère par défaut de la base. *buffers* est le nombre de buffer de base à allouer pour le cache serveur. Si il est passé à 0 ou omis, le serveur choisira de lui-même. *dialect* sélectionne le dialecte SQL pour les requêtes exécutées avec cette connexion, et par défaut, il utilise le meilleur dialecte disponible.

Si un deuxième appel est fait avec [ibase_connect\(\)](#), en passant les mêmes arguments, une nouvelle connexion ne sera pas ouverte, mais la connexion déjà ouverte sera retournée. La connexion sera fermée dès que le script se termine, à moins qu'elle ne soit fermée explicitement avec [ibase_close\(\)](#), durant le script.

Exemple [ibase_connect\(\)](#)

```
<?php
    $dbh = ibase_connect ($host, $username, $password);
    $stmt = 'SELECT * FROM tblname';
    $sth = ibase_query ($dbh, $stmt);
    while ($row = ibase_fetch_object ($sth)) {
        print $row->email . "\n";
    }
    ibase_close ($dbh);
?>
```

Note : *buffers* a été ajouté dans PHP4-RC2.

Note : *dialect* a été ajouté dans PHP4-RC2. Il n'est opérationnel qu'avec les versions InterBase 6 et plus

récentes.

Note : **role** a été ajouté dans PHP4–RC2. Il n'est opérationnel qu'avec les versions InterBase 5 et plus récentes.

Voir aussi: [ibase_pconnect\(\)](#).

10.29.2 [ibase_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_pconnect](#) (string **database**, string **username** , string **password** , string **charset** , int **buffers** , int **dialect** , string **role**)

[PHP 3>= 3.0.6, PHP 4]

[ibase_pconnect\(\)](#) se comporte similairement à [ibase_connect\(\)](#), avec deux différences majeures : la première est que, lors de la connexion, la fonction va essayer de trouver une connexion (perisitante) déjà ouverte. Si elle la trouve, cette dernière sera retournée, plutôt qu'une nouvelle connexion. Sinon, une nouvelle connexion sera ouverte. La deuxième est que la connexion ne sera pas fermée à la fin du script, mais restera ouverte pour utilisation ultérieure. ([ibase_close\(\)](#) ne fermera pas une connexion ouverte avec [ibase_pconnect\(\)](#). Ce type de lien est alors dit 'persistant'.

Note : **buffers** a été ajouté dans PHP4–RC2.

Note : **dialect** a été ajouté dans PHP4–RC2. Il n'est opérationnel qu'avec les versions InterBase 6 et plus récentes.

Note : **role** a été ajouté dans PHP4–RC2. Il n'est opérationnel qu'avec les versions InterBase 5 et plus récentes.

Voir aussi [ibase_connect\(\)](#) pour plus de détails sur les arguments de cette fonction.

10.29.3 [ibase_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_close](#) (int **connection_id**)

[PHP 3>= 3.0.6, PHP 4]

[ibase_close\(\)](#) ferme une connexion à une base de données Interbase. Cette fonction prend comme argument l'identifiant de connexion **connection_id** retourné par [ibase_connect\(\)](#). Si **connection_id** est omis, la dernière connexion ibase est fermée. Les transactions par défaut sont validées et les autres sont annulées.

10.29.4 [ibase_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_query](#) (int **link_identifieur** , string **query** , int **bind_args**)

[PHP 3>= 3.0.6, PHP 4]

Exécute une requête sur une base Interbase, et retourne un identifiant de résultat, à utiliser avec [ibase_fetch_row\(\)](#), [ibase_free_result\(\)](#) et/ou [ibase_free_query\(\)](#).

Note : Bien que ces fonctions supportent la liaison de variables avec des paramètres de requêtes, il n'y a pas d'intérêt spécial à les utiliser. Pour des exemples grandeur réelle, voyez [ibase_prepare\(\)](#) et [ibase_execute\(\)](#).

10.29.5 ibase_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ibase_fetch_row](#) (int *result_id*)

[PHP 3>= 3.0.6, PHP 4]

Retourne la prochaine ligne spécifiée dans le résultat obtenu de [ibase_query\(\)](#).

10.29.6 ibase_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [ibase_fetch_object](#) (int *result_id*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Lit une ligne dans une base Interbase et la place dans un pseudo objet. Cette fonction prend comme argument un identifiant de résultat obtenu de [ibase_query\(\)](#) ou [ibase_execute\(\)](#).

```
<php
    $dbh = ibase_connect ($host, $username, $password);
    $stmt = 'SELECT * FROM tblname';
    $sth = ibase_query ($dbh, $stmt);
while ($row = ibase_fetch_object ($sth)) {
    print $row->email . "\n";
}
ibase_close ($dbh);
?>
```

Voir aussi [ibase_fetch_row\(\)](#).

10.29.7 ibase_field_info

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ibase_field_info](#) (int *result*, int *field number*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Retourne un tableau contenant les informations sur un champs après une requête de SELECT. Le tableau contient les index name (nom), alias, relation, length (taille), type.

```
// helio@helio.com.br 08-Dec-2000 02:53
$rs=ibase_query("Select * from unetable");
$coln = ibase_num_fields($rs);
for ($i=0 ; $i < $coln ; $i++) {
    $col_info = ibase_field_info($rs, $i);
    echo "nom: ".$col_info['name']."\n";
    echo "alias: ".$col_info['alias']."\n";
    echo "relation: ".$col_info['relation']."\n";
    echo "taille: ".$col_info['length']."\n";
    echo "type: ".$col_info['type']."\n";
}
```

10.29.8 ibase_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_free_result](#)(int *result_identifieur*)

[PHP 3>= 3.0.6, PHP 4]

Libère un résultat créé par [ibase_query\(\)](#).

10.29.9 ibase_prepare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_prepare](#)(int *link_identifieur* , string *query*)

[PHP 3>= 3.0.6, PHP 4]

Prépare une requête pour l'exécuter (avec [ibase_execute\(\)](#)).

10.29.10 ibase_execute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_execute](#)(int *query* , int *bind_args*)

[PHP 3>= 3.0.6, PHP 4]

Exécute une requête préparée (et éventuellement liée) par [ibase_prepare\(\)](#). Cette fonction est beaucoup plus efficace que [ibase_query\(\)](#), si vous effectuez plusieurs fois la même requête, en ne changeant que quelques paramètres.

```

<?php
    $updates = array(
1 => 'Eric',
5 => 'Filip',
7 => 'Larry'
    );
    $query = ibase_prepare("UPDATE FOO SET BAR = ? WHERE BAZ = ?");
    while (list($baz, $bar) = each($updates)) {
        ibase_execute($query, $bar, $baz);
    }
?>

```

10.29.11 ibase_trans

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_trans](#)(int *trans_args* , int *link_identifieur*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Prépare une transaction

10.29.12 ibase_commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_commit](#)(int *link_identifieur* , int *trans_number*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Valide la transaction *trans_number*, qui a été préparée avec [ibase_trans\(\)](#).

10.29.13 ibase_rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_rollback](#)(int *link_identifieur* , int *trans_number*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Annule la transaction *trans_number* qui a été préparée avec [ibase_trans\(\)](#).

10.29.14 ibase_free_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_free_query](#)(int *query*)

[PHP 3>= 3.0.6, PHP 4]

Libère la mémoire réservée par une requête préparée par [ibase_prepare\(\)](#).

10.29.15 ibase_timefmt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_timefmt](#)(string *format* , int *columntype*)

[PHP 3>= 3.0.6, PHP 4]

Fixe le format des colonnes de type dates, heure et timestamp, retournées par les requêtes. En interne, les colonnes sont formatées par la fonction C strftime() : reportez vous à sa documentation pour connaître la structure de la chaîne de format. *columntype* est une des constantes suivantes : IBASE_TIMESTAMP, IBASE_DATE and IBASE_TIME. Si il est omis, la valeur par défaut est IBASE_TIMESTAMP, pour compatibilité ascendante.

```
<?php
    // Les colonnes TIME de InterBase 6 seront retournées avec
    // la forme '05 heures 37 minutes'.
ibase_timefmt("%H heures %M minutes", IBASE_TIME);
?>
```

Vous pouvez aussi modifier les formats par défaut avec les directives PHP `ibase.timestampformat`, `ibase.dateformat` et `ibase.timeformat`.

Note : *columntype* a été ajouté dans PHP 4.0. Il n'a aucun sens jusqu'à InterBase version 6 et plus récent.

Note : Une modification incompatible avec l'existant est apparue dans PHP 4.0 lorsque la directive PHP `ibase.timeformat` a été renommée `ibase.timestampformat` et les directives `ibase.dateformat` et `ibase.timeformat` ont été ajoutées, de manière à les adapter à leur fonction.

10.29.16 ibase_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ibase_num_fields](#) (int *result_id*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Retourne le nombre de lignes dans un résultat.

```

<?php
$dbh = ibase_connect ($host, $username, $password);
$stmt = 'SELECT * FROM tblname';
$sth = ibase_query ($dbh, $stmt);
if ( ibase_num_rows($sth) > 0 ) {
while ($row = ibase_fetch_object ($sth)) {
print $row->email . "\n";
}
} else {
die("Aucun résultat");
}
ibase_close ($dbh);
?>

```

Note : [ibase_timefmt\(\)](#) ne fonctionne pas encore sous PHP4.

10.29.17 ibase_errmsg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ibase_errmsg](#) (void)

[PHP 3>= 3.0.7, PHP 4 >= 4.0RC1]

Retourne une chaîne contenant les messages d'erreurs.

10.30 ICAP

[\[Notes en ligne\]](#)

Pour faire fonctionner ces fonctions, vous devez compiler PHP avec l'option `--with-icap`. Il vous faudra aussi la librairie icap. Récupérez la dernière version à <http://icap.chek.com/> et décompilez-la, puis installez la.

10.30.1 icap_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

stream [icap_open](#) (string *calendar*, string *username*, string *password*, string *options*)

[PHP 4 >= 4.0b4]

[icap_open\(\)](#) retourne un flot ICAP en cas de succès, FALSE en cas d'erreur.

[icap_open\(\)](#) ouvre une connexion ICAP au serveur de calendrier *calendar*. Si le paramètre *options* est spécifié, passe *options* à la boîte aux lettres(???).

10.30.2 icap_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_close](#)(int *icap_stream*, int *flags*)

[icap_close\(\)](#) ferme le flot ICAP *icap_stream*.

10.30.3 icap_fetch_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_fetch_event](#)(int *stream_id*, int *event_id*, int *options*)

[PHP 4 >= 4.0b4]

[icap_fetch_event\(\)](#) recherche un événement dans le calendrier spécifié par *id*.

Retourne un objet événement dont les attributs sont :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

10.30.4 icap_list_events

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [icap_list_events](#)(int *stream_id*, int *begin_date*, int *end_date*)

[PHP 4 >= 4.0RC1]

[icap_list_events\(\)](#) retourne un tableau d'identifiants d'événements, compris entre deux dates.

[icap_list_events\(\)](#) prend une date de début et une date de fin. Un tableau d'identifiants est retourné.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois

- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

10.30.5 icap_store_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [icap_store_event](#) (int *stream_id*, object *event*)
[PHP 4 >= 4.0b4]

[icap_store_event\(\)](#) enregistre un événement dans un calendrier ICAP.
Un événement est un objet avec les attributs suivants :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

10.30.6 icap_delete_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [icap_delete_event](#) (int *sream_id*, int *uid*)
[PHP 4 >= 4.0b4]

[icap_delete_event\(\)](#) efface l'événement d'identifiant *uid*.
Retourne TRUE.

10.30.7 icap_snooze

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [icap_snooze](#)(int *stream_id*, int *uid*)

[PHP 4 >= 4.0b4]

[icap_snooze\(\)](#) éteint l'alarme de l'événement identifié par l'UID *uid*.

Retourne TRUE.

10.30.8 icap_list_alarms

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [icap_list_alarms](#)(int *stream_id*, array *date*, array *time*)

[PHP 4 >= 4.0b4]

[icap_list_alarms\(\)](#) retourne un tableau d'identifiants, qui ont une alarme de prévue à la date *alarm_date*.

[icap_list_alarms\(\)](#) prend une date, et retourne un tableau d'identifiants.

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes

10.31 Informix

[\[Notes en ligne\]](#)

Les pilotes d'accès à Informix pour Online (ODS) 7.x, SE 7.x, Universal Server (IUS) 9.x et IDS 2000 sont implémentés dans "functions/ifx.ec" et "functions/php3_ifx.h". Le support ODS 7.x est plutôt complet, et accepte les colonnes de type BYTE et TEXT. Le support IUS 9.x est partiellement fini, de nouveaux types sont disponibles, mais SLOB et CLOB sont toujours en développement.

Note : Vous avez besoin d'une version de ESQL/C pour compiler le pilote PHP d'Informix. Les versions ESQL/C 7.2x sont utilisables. ESQL/C fait partie du SDK Informix Client.

Avant que vous ne lanciez le script "configure", assurez vous que la variable d'environnement "INFORMIXDIR" a été correctement paramétrée, et que \$INFORMIXDIR/bin est dans votre PATH.

Le script de configuration va détecter automatiquement les bibliothèques disponibles, et inclure les dossiers si vous lancer le script avec l'option "configure --with_informix=yes". Vous pouvez ignorer cette détection en spécifiant "IFX_LIBDIR", "IFX_LIBS" et "IFX_INCDIR" dans votre environnement. Le script de configuration va aussi essayer de détecter la version de votre serveur Informix. Il modifiera alors la condition de compilation "HAVE_IFX_IUS" si votre serveur Informix est d'une version plus récente que 9.00.

Note : Assurez vous que les variables d'environnement INFORMIXDIR et INFORMIXSERVER sont accessibles au pilote PHP, et que le dossier bin INFORMIX est aussi dans la variable PATH. Vous pouvez le voir en lançant un script qui contient un appel à [phpinfo\(\)](#) avant que vous ne commenciez à tester. La fonction [phpinfo\(\)](#) affiche une liste des variables d'environnement. Cela fonctionne aussi bien en mode

mod_php, qu'en mode CGI. Il vous faudra fixer les valeurs dans le script de démarrage d'Apache.

Les "Informix shared libraries" doivent aussi être accessibles au chargement (vérifiez `LD_LIBRARY_PATH` ou `ld.so.conf/ldconfig`).

Note : Les objets de type BLOBs sont normalement gérés par des identifiants de BLOB. Les requêtes de sélection retournent un identifiant de BLOB pour chaque colonne de type BYTE et TEXT. Vous pouvez en lire le contenu, avec des commandes de types "string_var = `ifx_get_blob($BLOB_id);`"; si vous souhaitez ramener le BLOB en mémoire (avec: "`ifx_blobinfile_mode(0);`"). Si vous préférez recevoir le contenu d'une colonne BLOB dans un fichier, utilisez [`ifx_blobinfile_mode\(\)`](#), et `ifx_get_blob($BLOB_id)` vous retournera le nom du fichier. Utilisez les fonctions habituelles d'accès aux fichiers pour lire son contenu.

Pour les requêtes INSERT/UPDATE, vous devez créer les identifiants de BLOB par vous même, avec la fonction [`ifx_create_blob\(\)`](#). Puis, vous placez l'identifiant de BLOB dans un tableau, et remplacez la colonne par un point d'interrogation. Pour les UPDATE/INSERT, vous êtes responsable du contenu du BLOB, avec la fonction [`ifx_update_blob\(\)`](#).

Le comportement par défaut des colonnes de type BLOB peut être modifié en affectant de nouvelles valeurs aux variables de configuration (même à la volée) :

Variable de configuration : `ifx.textasvarchar`

Variable de configuration : `ifx.byteasvarchar`

Fonctions à utiliser lors de l'exécution :

`ifx_textasvarchar(0)` : Utilise l'identifiant de BLOB avec des colonnes de type TEXT, dans les requêtes SELECT

`ifx_byteasvarchar(0)` : Utilise l'identifiant de BLOB avec des colonnes de type BYTE, dans les requêtes SELECT

`ifx_textasvarchar(1)` : Retourne les colonnes de type TEXT sous la forme de VARCHAR, sans utiliser les identifiants de BLOB dans les requêtes SELECT.

`ifx_byteasvarchar(1)` : Retourne les colonnes de type BYTE sous la forme de VARCHAR, sans utiliser les identifiants de BLOB dans les requêtes SELECT.

Variable de configuration : `ifx.BLOBinfile`

Fonctions à utiliser lors de l'exécution :

`ifx_blobinfile_mode(0)` : Retourne les colonnes de type BYTE en mémoire, l'identifiant de BLOB vous donnera accès au contenu.

`ifx_blobinfile_mode(1)` : Retourne les colonnes de type BYTE dans un fichier, l'identifiant de BLOB vous donnera accès au nom de ce fichier.

En affectant la valeur de 1 à `ifx_text/byteasvarchar`, vous pouvez utiliser les colonnes de type TEXT et BYTE dans les requêtes SELECT comme des champs VARCHAR (mais plus long). Etant donné la gestion des chaînes par PHP, cette technique conserve les données binaires. Les données retournées peuvent contenir n'importe quoi, et vous êtes responsable de la bonne manipulation de ces valeurs.

En affectant la valeur de 1 à [`ifx_blobinfile_mode\(\)`](#), utilisez le nom de fichier retourné par [`ifx_get_blob\(\)`](#) pour accéder au contenu du BLOB. Notez bien que vous êtes tenu responsable de l'effacement des fichiers temporaires, créés par Informix. Chaque nouvelle ligne lue sur le serveur va créer un nouveau fichier temporaire, pour chaque colonne de type BYTE.

L'emplacement des fichiers temporaire peut être modifié, grâce à la variable "blobdir", (par défaut, ".", c'est à dire, le dossier courant). Une valeur telle que `BLOBdir="tmpBLOB"` simplifiera le nettoyage des fichiers temporaires, accidentellement oubliés (les noms commencent tous par "blb").

Note : Elle peut être mise en place avec la variable de configuration.

`ifx.charasvarchar` : avec la valeur 1, les espaces de fin de champs seront automatiquement supprimés.

Note : Lorsque la variable de configuration `ifx.nullformat` (ou que la fonction [`ifx_nullformat\(\)`](#)) est à un, les colonnes contenant la valeur NULL retourneront la chaîne "NULL", et sinon, retourneront une chaîne vide. Cela vous permet de faire la différence entre les colonnes vides et celle qui contiennent la valeur NULL.

10.31.1 [ifx_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_connect](#) (string *database* , string *userid* , string *password*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_connect\(\)](#) retourne un identifiant de connexion, en cas de succès, et FALSE sinon.

[ifx_connect\(\)](#) établit une connexion à un serveur Informix. Tous les arguments sont optionnels, et, si ils viennent à manquer, les valeurs par défaut seront prises dans le fichier [7.1 Le fichier de configuration](#).

(ifx.default_host pour l'hôte par défaut) (Les bibliothèques Informix utiliseront la variable d'environnement **\$INFORMIXSERVER** si ifx.default_host n'est pas définie). ifx.default_user pour l'utilisateur, et ifx.default_password comme mot de passe (si aucun n'a été défini).

Si un deuxième appel à [ifx_connect\(\)](#) est fait avec les mêmes arguments, l'identifiant de connexion déjà ouvert sera retourné.

Le lien avec le serveur sera fermé dès que le script se termine, ce qui fait qu'il n'est pas nécessaire de terminer les connexions avec [ifx_close\(\)](#).

Voir aussi [ifx_pconnect\(\)](#) et [ifx_close\(\)](#).

Connection à un serveur Informix

```
<?php
$conn_id = ifx_pconnect ("mydb@ol_srv1", "imyself", "mypassword");
?>
```

10.31.2 [ifx_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_pconnect](#) (string *database* , string *userid* , string *password*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_pconnect\(\)](#) retourne un identifiant positif de connexion Informix, ou FALSE, en cas d'erreur.

[ifx_pconnect\(\)](#) se comporte de manière très similaire à [ifx_connect\(\)](#) avec deux différences importantes :

[ifx_pconnect\(\)](#) se comporte exactement comme [ifx_connect\(\)](#) lorsque PHP n'est pas un module Apache. Lors de la connexion, la fonction va chercher une connexion déjà ouverte avec le même hôte, le même nom d'utilisateur, et le même mot de passe. Si elle en trouve une, elle retournera un identifiant de cette connexion, au lieu d'en ouvrir une nouvelle.

Deuxièmement, la connexion au serveur SQL ne sera pas automatiquement refermée à la fin de l'exécution du script. Au contraire, le lien va rester ouvert ([ifx_close\(\)](#) ne fermera pas les connexions établies avec [ifx_pconnect\(\)](#)).

Ainsi, ce type de lien est appelé 'persistant'.

Voir aussi: [ifx_connect\(\)](#).

10.31.3 [ifx_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_close](#) (int *link_identifier*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_close\(\)](#) retourne toujours TRUE.

[ifx_close\(\)](#) ferme le lien au serveur de données Informix associé à l'identifiant de connexion *link_identifier*. Si l'identifiant du lien n'est pas spécifié, la dernière connexion est utilisée.

Notez qu'il n'est généralement pas besoin d'appeler cette fonction, car les connexions non persistantes seront automatiquement fermées.

[ifx_close\(\)](#) ne peut pas fermer une connexion ouverte avec [ifx_pconnect\(\)](#).

Voir aussi: [ifx_connect\(\)](#) et [ifx_pconnect\(\)](#).

Fermer une connexion Informix

```
<?php
$conn_id = ifx_connect ("mydb@ol_srv", "coucou", "cestmoi");
//... quelques requêtes diverses et variées ...
ifx_close($conn_id);
?>
```

10.31.4 ifx_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_query](#) (string *query*, int *link_identifier* , int *cursor_type* , mixed *blobidarray*)
[PHP 3>= 3.0.3, PHP 4]

[ifx_query\(\)](#) retourne un identifiant positif de résultat Informix en cas de succès, et FALSE en cas d'erreur.

L'entier de type "identifiant de résultat" est utilisé par 4 d'autres fonctions pour lire les résultats. Pour un exemple, reportez vous à [ifx_affected_rows\(\)](#) pour connaître le nombre de lignes affectées.

[ifx_query\(\)](#) envoie une requête au serveur actif courant, associé à l'identifiant de connexion *link_identifier*. Si *link_identifier* n'est pas fourni, la dernière connexion ouverte sera utilisée. Si aucune connexion n'a été ouverte, [ifx_query\(\)](#) va essayer d'en créer une, en appelant [ifx_connect\(\)](#).

Exécute la requête *query* sur la connexion *conn_id*. Pour les requêtes de type SELECT, un pointeur est déclaré, et ouvert. L'option *cursor_type* permet de choisir le type de pointeur, "scroll" et/ou "hold".

cursor_type accepte les deux valeurs séparées, et leur combinaison. Les requêtes d'autre type sont à exécution immédiate.

Le nombre de lignes affectées (estimé ou exact) est enregistré pour être lu avec [ifx_affected_rows\(\)](#).

Si vous avez une colonne de type BLOB (BYTE ou TEXT) dans une requête de modification, vous pouvez passer un paramètre *BLOBidarray* qui contiendra les identifiants des BLOB à modifier, et vous devrez remplacer cette colonne par un point d'interrogation (?) dans la requête.

Si le contenu d'une colonne de type TEXT (ou BYTE) vous pouvez aussi utiliser les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#). Cela vous permettra d'utiliser les colonnes TEXT (ou BYTE) comme des colonnes de type VARCHAR (mais plus long, tout de même), et vous n'aurez pas besoin de l'identifiant de BLOB.

Avec les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#) (valeurs par défaut), les requêtes SELECT retourneront des identifiants de BLOB. Cet identifiant peut être une chaîne ou un fichier, suivant la configuration (voir plus loin).

Voir aussi : [ifx_connect\(\)](#).

Afficher toutes les lignes de la table "ordres" sous la forme html

```
ifx_textasvarchar(1);          // Utilisation du mode "text mode" pour les BLOBs
$res_id = ifx_query("select * from orders", $conn_id);
if (! $res_id) {
    printf("Impossible de selectionner des lignes dans : %s\n<br>%s<br>\n", ifx_error());
    ifx_errormsg();
}
```



```
die;
}
ifx_htmltbl_result($res_id, "border=\"1\"");
ifx_free_result($res_id);
```

Insertion de valeurs dans la table "catalogue"

```
<?php
// créer un identifiant de BLOB pour une colonne de type BYTE et une de typ
$textid = ifx_create_blob(0, 0, "Colonne Text en mémoire");
$byteid = ifx_create_blob(1, 0, "Colonne Byte en mémoire");
// Stocke l'identifiant du BLOB dans le tableau BLOBid
$BLOBidarray[] = $textid;
$BLOBidarray[] = $byteid;
// launch query
$query = "insert into catalog (stock_num, manu_code, " .
        "cat_descr,cat_picture) values(1,'HRO',?,?)";
$res_id = ifx_query($query, $conn_id, $BLOBidarray);
if (! $res_id) {
    ... erreur ...
}
// libération du résultat
ifx_free_result($res_id);
?>
```

10.31.5 ifx_prepare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_prepare](#)(string *query*, int *conn_id*, int *cursor_def*, mixed *blobidarray*)
[PHP 3>= 3.0.4, PHP 4]

[ifx_prepare\(\)](#) retourne un entier identifiant de résultat *result_id* à utiliser avec [ifx_do\(\)](#). Modifie la valeur de *affected_rows*, pour accès ultérieur avec [ifx_affected_rows\(\)](#).

Prépare la requête *query* sur la connexion *conn_id*. Pour les requêtes de type "select-type" un pointeur de résultat est déclaré et ouvert. L'option *cursor_type* permet de choisir le type de pointeur : "scroll" et/ou "hold". Les valeurs peuvent être combinées ensemble (IFX_SCROLL, IFX_HOLD).

Le nombre de ligne affectés (estimé ou exact) est enregistré, pour être lu avec la fonction

[ifx_affected_rows\(\)](#).

Si vous avez une colonne de type BLOB (BYTE ou TEXT) dans une requête de modification, vous pouvez passer un paramètre *BLOBidarray* qui contiendra les identifiants des BLOB à modifier, et vous devrez remplacer cette colonne par un point d'interrogation (?) dans la requête.

Si le contenu d'une colonne de type TEXT (ou BYTE) vous pouvez aussi utiliser les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#). Cela vous permettra d'utiliser les colonnes TEXT (ou BYTE) comme des colonnes de type VARCHAR (mais plus long, tout de même), et vous n'aurez pas besoin de l'identifiant de BLOB.

Avec les fonctions [ifx_textasvarchar\(\)](#) et [ifx_byteasvarchar\(\)](#) (valeurs par défaut), les requêtes SELECT retourneront des identifiants de BLOB. Cet identifiant peut être une chaîne ou un fichier, suivant la configuration (voir plus loin).

Voir aussi [ifx_do\(\)](#).

10.31.6 ifx_do

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_do](#)(int *result_id*)

[PHP 3>= 3.0.4, PHP 4]

[ifx_do\(\)](#) retourne TRUE en cas de succès, FALSE en cas d'erreur.

Exécute une requête qui a déjà été préparée, ou crée un pointeur pour cela.

Ne libère pas *result_id* en cas d'erreur.

De plus, elle fixe la valeur de *result_id* pour accès ultérieur par [ifx_affected_rows\(\)](#).

Voir aussi : [ifx_prepare\(\)](#). Il y a un exemple.

10.31.7 ifx_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ifx_error](#)

[PHP 3>= 3.0.3, PHP 4]

[ifx_error\(\)](#) retourne le code d'erreur de la dernière requête Informix. Les codes d'erreur Informix (SQLSTATE & SQLCODE) formaté comme suit :

x [SQLSTATE = aa bbb SQLCODE=cccc]

avec x = space : aucune erreur

E : erreur

N : il n'y a plus d'informations

W : Alerte

? : Undéfinie

Si le caractère vaut autre chose qu'un espace, SQLSTATE et SQLCODE décrit l'erreur avec plus de détails.

Reportez vous au manuel Informix pour trouver la description de SQLSTATE et SQLCODE

Retourne une chaîne avec un caractères, décrivant le résultat général de la commande, et aussi SQLSTATE et SQLCODE associé à la plus récente requête SQL exécutée. Le format de la chaîne est "(char)

[SQLSTATE=(deux chiffres) (trois chiffres) SQLCODE=(un chiffre)]". Le premier caractère peut être ' ' (espace) (succès), 'W' (Alerte), 'E' (une erreur est survenue durant le traitement) ou 'N' (aucune donnée de retour).

Voir aussi: [ifx_errormsg\(\)](#).

10.31.8 ifx_errormsg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ifx_errormsg](#)(int *errorcode*)

[PHP 3>= 3.0.4, PHP 4]

[ifx_errormsg\(\)](#) retourne le plus récent message d'erreur ou, lorsque l'option *errorcode* est présent, le message d'erreur associé à *errorcode*.

Voir aussi: [ifx_error\(\)](#).

```
printf("%s\n<br>", ifx_errormsg(-201));
```

10.31.9 ifx_affected_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_affected_rows](#)(int *result_id*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_affected_rows\(\)](#) retourne le nombre de lignes affectées par la requête associée à *result_id*.

result_id est un identifiant valide de résultat retourné par [ifx_query\(\)](#) ou [ifx_prepare\(\)](#).

Pour les INSERT, UPDATE et DELETE, ce nombre est le nombre exact de lignes affectées (sqlerrd[2]). Pour les SELECT, ce n'est qu'une estimation (sqlerrd[0]). Ne vous y fiez pas.

[ifx_affected_rows\(\)](#) est très pratique après [ifx_prepare\(\)](#) pour limiter la taille des résultats.

Voir aussi : [ifx_num_rows\(\)](#).

Nombre de lignes affectées

```

<?php
$rid = ifx_prepare ("select * from emp
where name like " . $name, $connid);
if (! $rid) {
    //... erreur ...
}
$rowcount = ifx_affected_rows ($rid);
if ($rowcount > 1000) {
    printf ("Trop de lignes trouvées (%d)\n<br>", $rowcount);
    die ("Essayez avec une autre requête. <br>\n");
}
?>

```

10.31.10 ifx_getsqlca

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_getsqlca](#)(int *result_id*)

[PHP 3>= 3.0.8, PHP 4]

[ifx_getsqlca\(\)](#) retourne une pseudo-ligne (tableau associatif) avec sqlca.sqlerrd[0] à sqlca.sqlerrd[5] après la requête associée *result_id*.

result_id est un identifiant valide de résultat retourné par [ifx_query\(\)](#) ou [ifx_prepare\(\)](#).

Pour les requêtes INSERT, UPDATE et DELETE, les valeurs retournées sont celles fixées par le serveur après avoir exécuté la requête. Cela donne accès au nombre de ligne affectées, ainsi qu'au numéro de série d'insertion. Pour les requêtes de type SELECT, les valeurs retournées sont celles qui ont été préparées.

Utiliser cette fonction économise l'exécution d'une requête "select dbinfo('sqlca.sqlerrdx')", étant donné qu'elle retourne les valeurs qui ont été sauveés par le pilote ifx au moment approprié.

Lire les valeurs de sqlca.sqlerrd[x]

```

<?php
/* On suppose que la première colonne d'une table 'quelconque' est un numéro de série */
$qid = ifx_query("insert into sometable values(0, '2nd column', 'another column' ", $connid);
if (! $qid) {
    ... erreur ...
}
$sqlca = ifx_getsqlca ($qid);
$serial_value = $sqlca["sqlerrd1"];

```

```
echo "Le numéro de série de la valeur insérée est : " . $serial_value<br>\n";
?>
```

10.31.11 ifx_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_fetch_row](#) (int *result_id*, mixed *position*)
[PHP 3>= 3.0.3, PHP 4]

[ifx_fetch_row\(\)](#) retourne un tableau associatif qui contient la ligne retournée, ou FALSE si il ne reste plus de lignes à lire, ou si il a eu une erreur.

Les colonnes de types BLOB sont retournées sous la forme d'un identifiant à utiliser avec [ifx_get_blob\(\)](#) à moins que vous n'ayez utilisé la fonction [ifx_textasvarchar\(\)](#) ou [ifx_byteasvarchar\(\)](#), et dans ce cas, les BLOBs seront retournés sous forme de chaîne. Retourne FALSE en cas d'erreur.

result_id est un identifiant valide de résultat, retourné par [ifx_query\(\)](#) ou [ifx_prepare\(\)](#) (Requêtes SELECT seulement !).

position est un paramètre optionnel, pour une opération de lecture d'informations sur un pointeur de type "scroll": "NEXT", "PREVIOUS", "CURRENT", "FIRST", "LAST" ou encore un nombre. Si vous spécifiez un nombre, la ligne d'index absolu sera retournée. Ce paramètre est optionnel, et ne fonctionne qu'avec les pointeurs de type "scroll".

[ifx_fetch_row\(\)](#) retourne une ligne de données d'un résultat associé à l'identifiant de résultat *result_id*. La ligne est retournée sous la forme d'un tableau associatif.

Les appels ultérieurs à [ifx_fetch_row\(\)](#) retourneront la ligne suivante, ou FALSE si il n'y a plus de ligne.

Exemple avec ifx_fetch_row()

```
<?php
$rid = ifx_prepare ("select * from emp where name like " . $name,
                  $connid, IFX_SCROLL);

if (! $rid) {
    ... erreur ...
}
$rowcount = ifx_affected_rows($rid);
if ($rowcount > 1000) {
    printf ("Trop de lignes dans le résultats. (%d)\n<br>", $rowcount);
    die ("Recommencez votre requête. <br>\n");
}
if (! ifx_do ($rid)) {
    ... erreur ...
}
$row = ifx_fetch_row ($rid, "NEXT");
while (is_array($row)) {
    for(reset($row); $fieldname=key($row); next($row)) {
        $fieldvalue = $row[$fieldname];
        printf ("%s = %s,", $fieldname, $fieldvalue);
    }
    printf("\n<br>");
    $row = ifx_fetch_row ($rid, "NEXT");
}
ifx_free_result ($rid);
?>
```

10.31.12 ifx_htmltbl_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_htmltbl_result](#)(int *result_id*, string *html_table_options*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_htmltbl_result\(\)](#) lit toutes les lignes d'un tableau, et la met sous la forme d'un tableau HTML, ou FALSE en cas d'erreur.

Affiche les lignes avec des balises HTML. Le second argument permet de modifier les options de table.

Affichage sous la forme d'une table HTML

```

<?php
$rid = ifx_prepare ("select * from emp where name like " . $name,
                  $connid, IFX_SCROLL);
if (! $rid) {
    ... erreur ...
}
$rowcount = ifx_affected_rows ($rid);
if ($rowcount > 1000) {
    printf ("Trop de lignes dans le résultat : (%d)\n<br>", $rowcount);
    die ("Recommencez votre requête <br>\n");
}
if (! ifx_do($rid) {
    ... erreur ...
}
ifx_htmltbl_result ($rid, "border=\"2\"");
ifx_free_result($rid);
?>

```

10.31.13 ifx_fieldtypes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_fieldtypes](#)(int *result_id*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_fieldtypes\(\)](#) retourne un tableau associatif avec les noms des champs comme clés, et les types SQL comme valeur. En cas d'erreur, retourne FALSE.

Nom de champs et type SQL.

```

<?php
$types = ifx_fieldtypes ($resultid);
if (! isset ($types)) {
    ... erreur ...
}
for ($i = 0; $i < count($types); $i++) {
    $fname = key($types);
    printf("%s : \t type = %s\n", $fname, $types[$fname]);
    next($types);
}
?>

```

10.31.14 ifx_fieldproperties

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ifx_fieldproperties](#) (int *result_id*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_fieldproperties\(\)](#) retourne un tableau associatif avec les nom des champs comme clé, et les données de propriétés des champs comme valeur. Retourne FALSE en cas d'erreur.

Retourne les propriétés Informix SQL pour tous les champs d'une requête, sous la forme d'un tableau associatif. Les propriétés sont présentées sous la forme : "SQLTYPE;longueur

;précision;échelle;ISNULLABLE" avec SQLTYPE qui représente le type de données Informix tel que "SQLVCHAR" et ISNULLABLE = "Y" ou "N" (le champs peut contenir NULL ou pas : Oui ou Non).

Exemple avec ifx_fieldproperties()

```

<?php
$properties = ifx_fielddtypes ($resultid);
if (! isset($properties)) {
    ... erreur ...
}
for ($i = 0; $i < count($properties); $i++) {
    $fname = key ($properties);
    printf ("%s:\t type = %s\n", $fname, $properties[$fname]);
    next ($properties);
}
?>

```

10.31.15 ifx_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_num_fields](#) (int *result_id*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_num_fields\(\)](#) retourne le nombre de colonnes dans la requête *result_id* ou FALSE en cas d'erreur.

Après avoir préparé ou exécuté une requête, cette fonction retourne le nombre de colonne dans la requête.

10.31.16 ifx_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_num_rows](#) (int *result_id*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_num_rows\(\)](#) compte le nombre de ligne déjà lues dans le résultat *result_id* après [ifx_query\(\)](#) ou [ifx_do\(\)](#).

10.31.17 ifx_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_free_result](#) (int *result_id*)

[PHP 3>= 3.0.3, PHP 4]

[ifx_free_result\(\)](#) libère les ressources prises par le résultat *result_id*. Retourne FALSE en cas d'erreur.

[10.31.18 ifx_create_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_create_char](#)(string *param*)

[PHP 3>= 3.0.6, PHP 4]

[ifx_create_char\(\)](#) crée un objet char. *param* sera le contenu de l'objet.

[10.31.19 ifx_free_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_free_char](#)(int *bid*)

[PHP 3>= 3.0.6, PHP 4]

[ifx_free_char\(\)](#) supprime l'objet char *bid*. Retourne FALSE en cas d'erreur, et sinon TRUE.

[10.31.20 ifx_update_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_update_char](#)(int *bid*, string *content*)

[PHP 3>= 3.0.6, PHP 4]

[ifx_update_char\(\)](#) modifie le contenu de l'objet char repéré par son identifiant *bid*. *content* est une chaîne avec les nouvelles données. Retourne. FALSE en cas d'erreur, et sinon, TRUE.

[10.31.21 ifx_get_char](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_get_char](#)(int *bid*)

[PHP 3>= 3.0.6, PHP 4]

[ifx_get_char\(\)](#) retourne le contenu de l'objet associé à l'identifiant *bid*.

[10.31.22 ifx_create_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_create_blob](#)(int *type*, int *mode*, string *param*)

[PHP 3>= 3.0.4, PHP 4]

[ifx_create_blob\(\)](#) crée un objet BLOB.

type: 1 = TEXT, 0 = BYTE

mode: 0 = L'objet BLOB place le contenu en mémoire ; 1 = L'objet BLOB place le contenu dans un fichier.

param: Si mode = 0: pointeur du contenu, si mode = 1: pointeur vers un fichier.

Retourne FALSE en cas d'erreur, et sinon, un identifiant de BLOB.

[10.31.23 ifx_copy_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_copy_blob](#)(int *bid*)
[PHP 3>= 3.0.4, PHP 4]

[ifx_copy_blob\(\)](#) retourne FALSE en cas d'erreur, et sinon, l'identifiant du nouvel objet.

[10.31.24 ifx_free_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_free_blob](#)(int *bid*)
[PHP 3>= 3.0.4, PHP 4]

[ifx_free_blob\(\)](#) supprime l'objet BLOB *bid*. Retourne FALSE en cas d'erreur, et sinon TRUE.

[10.31.25 ifx_get_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifx_get_blob](#)(int *bid*)
[PHP 3>= 3.0.4, PHP 4]

[ifx_get_blob\(\)](#) retourne le contenu de l'objet BLOB associé à *bid*.

[10.31.26 ifx_update_blob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ifx_update_blob](#)(int *bid*, string *content*)
[PHP 3>= 3.0.4, PHP 4]

[ifx_update_blob\(\)](#) modifie le contenu de l'objet BLOB repéré par son identifiant *bid*. *content* est une chaîne contenant les nouvelles données. Retourne FALSE en cas d'erreur, et sinon, TRUE.

[10.31.27 ifx_blobinfile_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_blobinfile_mode](#)(int *mode*)
[PHP 3>= 3.0.4, PHP 4]

[ifx_blobinfile_mode\(\)](#) modifie le mode par défaut des objets BLOB pour toutes les requêtes SELECT. Mode "0" chargera les BLOB de type Byte en mémoire ; Mode "1" sauvera les BLOB de type Byte dans un fichier.

[10.31.28 ifx_textasvarchar](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_textasvarchar](#)(int *mode*)

[PHP 3>= 3.0.4, PHP 4]

[ifx_textasvarchar\(\)](#) modifie le mode par défaut des objets TEXT. Le mode "0" retournera un identifiant de BLOB et le mode "1" retourne le BLOB sous la forme d'un (gros) varchar.

10.31.29 ifx_byteasvarchar

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_byteasvarchar](#)(int *mode*)

[PHP 3>= 3.0.4, PHP 4]

[ifx_byteasvarchar\(\)](#) modifie le mode par défaut des objets BYTE. Le mode "0" retournera l'identifiant de BLOB, et le mode "1" retournera le contenu du text sous la forme d'un VARCHAR.

10.31.30 ifx_nullformat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ifx_nullformat](#)(int *mode*)

[PHP 3>= 3.0.4, PHP 4]

[ifx_nullformat\(\)](#) modifie le mode par défaut de lecture des valeurs. Le mode "0" retourne "", et le mode "1" retourne "NULL".

10.31.31 ifxus_create_slob

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_create_slob](#)(int *mode*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_create_slob\(\)](#) crée un objet SLOB et l'ouvre. Les modes valides sont : 1 = LO_RDONLY, 2 = LO_WRONLY, 4 = LO_APPEND, 8 = LO_RDWR, 16 = LO_BUFFER, 32 = LO_NOBUFFER -> ou une combinaison des précédents. Vous pouvez aussi utiliser les constantes suivantes : IFX_LO_RDONLY, IFX_LO_WRONLY etc. Retourne FALSE en cas d'erreur, et sinon, l'identifiant de l'objet SLOB.

10.31.32 ifx_free_slob

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_free_slob](#)(int *bid*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_free_slob\(\)](#) supprime un objet SLOB. *bid* est l'identifiant de l'objet SLOB. Retourne FALSE en cas d'erreur, et sinon TRUE.

10.31.33 ifxus_close_slob

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_close_slob](#)(int *bid*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_close_slob\(\)](#) ferme l'objet SLOB représenté par son identifiant *bid*. Retourne FALSE en cas d'erreur, et sinon, TRUE.

[10.31.34 ifxus_open_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_open_slob](#)(long *bid*, int *mode*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_open_slob\(\)](#) ouvre un objet SLOB. *bid* est un identifiant d'objet SLOB. Les modes valides sont : 1 = LO_RDONLY, 2 = LO_WRONLY, 4 = LO_APPEND, 8 = LO_RDWR, 16 = LO_BUFFER, 32 = LO_NOBUFFER → ou une combinaison des valeurs précédentes. Retourne FALSE en cas d'erreur, et sinon, l'identifiant du nouvel objet.

[10.31.35 ifxus_tell_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_tell_slob](#)(long *bid*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_tell_slob\(\)](#) retourne le fichier courant, ou la position courante d'un objet SLOB ouvert. *bid* est un identifiant d'objet SLOB. Retourne FALSE en cas d'erreur, et sinon, la position du pointeur de fichier.

[10.31.36 ifxus_seek_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_seek_slob](#)(long *bid*, int *mode*, long *offset*)

[PHP 3>= 3.0.4, PHP 4]

@xref{function.ifxus-seek-slob , , ifxus_seek_slob()} modifie le fichier courant, ou la position du pointeur de fichier, pour un objet SLOB ouvert. *bid* est un identifiant d'objet SLOB. Les modes valides sont : 0 = LO_SEEK_SET, 1 = LO_SEEK_CUR, 2 = LO_SEEK_END et *offset* est un octet d'offset. Retourne FALSE en cas d'erreur, et sinon, la position du pointeur de fichier.

[10.31.37 ifxus_read_slob](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_read_slob](#)(long *bid*, long *nbytes*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_read_slob\(\)](#) lit *nbytes* octets de l'objet SLOB *bid*. *bid* est un identifiant d'objet SLOB existant, et *nbytes* est le nombre d'octets à lire. Retourne FALSE en cas d'erreur, et sinon, une chaîne de caractères.

10.31.38 ifxus_write_slob

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ifxus_write_slob\(\)](#) (long *bid*, string *content*)

[PHP 3>= 3.0.4, PHP 4]

[ifxus_write_slob\(\)](#) écrit une chaîne dans un objet SLOB. *bid* est un identifiant d'objet SLOB et *content* sont les données à écrire. Retourne FALSE en cas d'erreur, et sinon, le nombre d'octets écrits.

10.32 Images

[\[Notes en ligne\]](#)

Vous pouvez utiliser les fonctions PHP pour obtenir les tailles des images aux formats *JPEG*, *GIF*, *PNG* et *SWF*, et si vous avez la librairie GD (disponible à <http://www.boutell.com/gd/>) vous pourrez aussi créer et manipuler ces images.

Les formats des images que vous pourrez manipuler dépend de la version de GD que vous installerez, et de toute autre librairie dont GD a besoin pour accéder à ces images. Les versions antérieures à la version 1.6 supportent le *GIF*, mais pas le *PNG*. Pour les versions plus récentes, c'est le contraire.

Pour accéder aux images en *JPEG*, vous devez installer la librairie jpeg-6b (disponible à <ftp://ftp.uu.net/graphics/jpeg/>), puis, recompiler GD pour qu'elle utilise jpeg-6b. Vous devrez aussi compiler PHP avec `--with-jpeg-dir=/path/to/jpeg-6b`.

Pour ajouter le support des polices Type 1, vous devez installer t1lib (disponible à <ftp://ftp.neuroinformatik.ruhr-uni-bochum.de/pub/software/t1lib/>), puis ajouter l'option `--with-t1lib[=dir]`.

10.32.1 getimagesize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getimagesize\(\)](#) (string *filename*, array *imageinfo*)

[PHP 3, PHP 4]

[getimagesize\(\)](#) va déterminer la taille des images de type *GIF*, *JPG*, *PNG* ou *SWF* et en retourner les dimensions avec le type d'image, et une chaîne type "height/width", à placer dans une balise *HTML* ou *IMG* normale.

[getimagesize\(\)](#) retourne un tableau de 4 éléments. L'index 0 contient la largeur. L'index 1 contient la longueur. L'index 2 contient le type de l'image : 1 = *GIF*, 2 = *JPG*, 3 = *PNG*. L'index 3 contient la chaîne à placer dans les balises *HTML* : "height=xxx width=xxx".

getimagesize()

```
<?php
$size = GetImageSize("img/flag.jpg");
?>
<IMG SRC="img/flag.jpg"
<?php
echo $size[3];
?>>
```

getimagesize() avec une url

```
<?php
$size = getimagesize("http://www.php.net/gifs/logo.gif");
?gt;
```

Le paramètre optionnel *imageinfo* permet d'extraire des informations supplémentaires du fichier image. Actuellement, cette option va retourner différents marqueurs **JPG APP** dans un tableau associatif. Certains programmes utilisent ces marqueur APP pour préciser les informations dans les balises HTML. Un marqueur commun est le marqueur APP13, décrit à <http://www.iptc.org/>. Vous pouvez utiliser la fonction [iptcparse\(\)](#) pour analyser ce marqueur, et obtenir des informations intelligibles.

getimagesize() qui retourne iptc

```
<?php
$size = getimagesize("testimg.jpg",&$info);
if (isset($info["APP13"])) {
    $iptc = iptcparse($info["APP13"]);
    var_dump($iptc);
}
?>
```

Note : [getimagesize\(\)](#) ne requiert par la bibliothèque GD.

Note : Le support URL a été ajouté en PHP 4.0.5.

[10.32.2 imagearc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagearc](#)(int *im*, int *cx*, int *cy*, int *w*, int *h*, int *s*, int *e*, int *col*)

[PHP 3, PHP 4]

[imagearc\(\)](#) dessine une ellipse partielle, centrée sur *cx*, *cy* (le coin en haut à gauche est l'origine (0,0)) dans l'image référencée par *im*. *w* et *h* spécifient la largeur et la hauteur de l'ellipse, tandis que le début et la fin de l'arc sont donnés en degrés, par les arguments *s* et *e*.

[10.32.3 imagechar](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagechar](#)(int *im*, int *font*, int *x*, int *y*, string *c*, int *col*)

[PHP 3, PHP 4]

[imagechar\(\)](#) dessine le premier caractère de la chaîne *c* dans l'image *id* avec le coin supérieur gauche placé à la position *x,y* (le coin en haut à gauche est l'origine (0,0)) avec la couleur *col*. Si la police est 1, 2, 3, 4 ou 5, une police intégrée sera utilisée (plus le chiffre est grand, plus grande est la police).0

Voir aussi [imageloadfont\(\)](#).

[10.32.4 imagecharup](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecharup](#)(int *im*, int *font*, int *x*, int *y*, string *c*, int *col*)

[PHP 3, PHP 4]

[`imagecharup\(\)`](#) dessine le premier caractère de la chaîne *c* dans l'image *id* avec le coin supérieur gauche placé à la position *x,y* (le coin en haut à gauche est l'origine (0,0)), avec la couleur *col*. Si la police est 1, 2, 3, 4 ou 5, une police intégrée sera utilisée (plus le chiffre est grand, plus grande est la police).

Voir aussi [`imageloadfont\(\)`](#).

[10.32.5 imagecolorallocate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`imagecolorallocate\(\)`](#) (int *im*, int *red*, int *green*, int *blue*)

[PHP 3, PHP 4]

[`imagecolorallocate\(\)`](#) retourne un identifiant de couleur, représentant la couleur composée avec les couleurs RGB (*red*, *green*, *blue*). L'argument *im* est le résultat de la fonction [`imagecreate\(\)`](#). [`imagecolorallocate\(\)`](#) doit être appelée pour créer chaque couleur qui sera représentée par *im*.

```
<?php
$white = imagecolorallocate($im, 255,255,255);
$black = imagecolorallocate($im, 0,0,0);
?>
```

[10.32.6 imagecolordeallocate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`imagecolordeallocate\(\)`](#) (int *im* , int *index*)

[PHP 3>= 3.0.6, PHP 4]

[`imagecolordeallocate\(\)`](#) désalloue une couleur précédemment allouée avec la fonction [`imagecolorallocate\(\)`](#).

```
<?php
$white = imagecolorallocate($im, 255, 255, 255);
imagecolordeallocate($im, $white);
?>
```

[10.32.7 imagecolorat](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`imagecolorat\(\)`](#) (int *im*, int *x*, int *y*)

[PHP 3, PHP 4]

[`imagecolorat\(\)`](#) retourne l'index de la couleur du pixel situé aux coordonnées (*x*, *y*), dans l'image *im*.

Voir aussi [`imagecolorset\(\)`](#) et [`imagecolorsforindex\(\)`](#).

[10.32.8 imagecolorclosest](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorclosest](#) (int *im*, int *red*, int *green*, int *blue*)

[PHP 3, PHP 4]

[imagecolorclosest\(\)](#) retourne l'index de la couleur de la palette qui est la plus proche de la valeur RGB passée. La "distance" entre la couleur souhaitée et les couleurs de la palette est calculée en considérant l'espace RGB comme un espace à 3 dimensions.

Voir aussi [imagecolorexact\(\)](#).

[10.32.9 imagecolorexact](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorexact](#) (int *im*, int *red*, int *green*, int *blue*)

[PHP 3, PHP 4]

[imagecolorexact\(\)](#) retourne l'index de la couleur spécifiée dans la palette de l'image *im*.

Si la couleur n'existe pas dans cette palette, retourne -1.

Voir aussi [imagecolorclosest\(\)](#).

[10.32.10 imagecolorresolve](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorresolve](#) (int *im*, int *red*, int *green*, int *blue*)

[PHP 3 >= 3.0.2, PHP 4]

[imagecolorresolve\(\)](#) retourne un index de couleur à tous les coups. Soit il arrive à trouver la couleur demandée dans la palette, soit il recherche la couleur la plus proche.

Voir aussi [imagecolorclosest\(\)](#).

[10.32.11 imagegammacorrect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagegammacorrect](#) (int *im* , double *inputgamma* , double *outputgamma*)

[PHP 3 >= 3.0.13, PHP 4 >= 4.0.0]

[imagegammacorrect\(\)](#) applique une correction gamma au flot d'image GD *im*. Le facteur d'entrée est *inputgamma*, et le facteur de sortie est *outputgamma*.

[10.32.12 imagecolorset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [imagecolorset](#) (int *im*, int *index*, int *red*, int *green*, int *blue*)

[PHP 3, PHP 4]

[imagecolorset\(\)](#) permet d'attribuer à un index d'une palette une couleur spécifique. C'est une fonction très pratique pour effectuer du remplissage de couleur sans le faire réellement.

Voir aussi [imagecolorat\(\)](#).

[10.32.13 imagecolorsforindex](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagecolorsforindex](#) (int *im*, int *index*)

[PHP 3, PHP 4]

[imagecolorsforindex\(\)](#) retourne un tableau associatif avec les couleur rouge (red) , vert (green), bleu (blue) qui contiennent les valeurs de la couleur correspondante.

Voir aussi [imagecolorat\(\)](#) et [imagecolorexact\(\)](#).

[10.32.14 imagecolorstotal](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolorstotal](#) (int *im*)

[PHP 3, PHP 4]

[imagecolorstotal\(\)](#) retourne le nombre de couleur de la palette.

Voir aussi [imagecolorat\(\)](#) et [imagecolorsforindex\(\)](#).

[10.32.15 imagecolortransparent](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecolortransparent](#) (int *im*, int *col*)

[PHP 3, PHP 4]

[imagecolortransparent\(\)](#) permet de choisir la couleur transparente d'une image, et de lui donner la valeur de col. *im* est un identifiant d'image, retourné par [imagecreate\(\)](#) et *col* est un identifiant de couleur retourné par [imagecolorallocate\(\)](#).

L'identifiant de la nouvelle (ou courante) couleur transparente est retourné.

[10.32.16 imagecopy](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecopy](#) (int *dst_im* , int *src_im* , int *dst_x* , int *dst_y* , int *src_x* , int *src_y* , int *src_w* , int *src_h*)

[PHP 3>= 3.0.6, PHP 4]

Copie une partie de l'image *src_im* sur l'image de destination *dst_im*, en commençant aux coordonnées *src_x*, *src_y* et sur la largeur de *src_w* et la hauteur de *src_h*. La portion ainsi définie sera copiée et placée aux coordonnées *dst_x* et *dst_y*.

[10.32.17 imagecopyresized](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecopyresized](#) (int *dst_im*, int *src_im*, int *dstX*, int *dstY*, int *srcX*, int *srcY*, int *dstW*, int *dstH*, int *srcW*, int *srcH*)

[PHP 3, PHP 4]

[imagecopyresized\(\)](#) copie une partie rectangulaire d'une image dans une autre image de destination. *dst_im* est l'image de destination, *src_im* est l'image source. Si les dimensions de la source et de la destination ne sont pas égales, un étirement adéquat est effectué pour faire correspondre les deux. Les coordonnées fournies se repèrent par rapport au coin supérieur gauche. Cette fonction peut être utilisée pour recopier des régions à l'intérieur d'une même image, si *dst_im* et *src_im* sont identiques : mais si les régions se chevauchent, le résultat risque d'être incohérent.

10.32.18 imagecreate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreate](#)(int *x_size*, int *y_size*)

[PHP 3, PHP 4]

[imagecreate\(\)](#) retourne un identifiant d'image représentant une image vide, de largeur *x_size* et longueur *y_size*.

10.32.19 imagecreatefromgif

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefromgif](#)(string *filename*)

[PHP 3, PHP 4]

[imagecreatefromgif\(\)](#) retourne un identifiant d'image qui représente l'image obtenue à partir du fichier dont le nom est donné.

[imagecreatefromgif\(\)](#) retourne une chaîne vide en cas d'échec. Il va aussi retourner une erreur qui va afficher un lien brisé dans un navigateur. Pour simplifier le débogage, utilisez le code suivant, qui retourne une erreur **GIF** :

Exemple de gestion des erreurs durant création d'image (gracieusement offert par vic@zysmsys.com)

```
<?php
function loadgif($imgname)
{
    $im = @imagecreatefromgif($imgname); /* Tentative d'ouverture */
    if ($im == "") { /* échec ? */
        $im = ImageCreate(150,30); /* Crée une image vide */
        $bgc = ImageColorAllocate($im,255,255,255);
        $ctc = ImageColorAllocate($im,0,0,0);
        imagefilledrectangle($im,0,0,150,30,$bgc);
        imagestring($im,1,5,5,"Erreur lors du chargement du fichier $imgname",$ctc);
        /* Affiche un message d'erreur */
    }
    return $im;
}
?>
```

Note : Etant donné que toutes les fonctions de gestion des **GIF** ont été supprimées de la bibliothèque GD version 1.6, cette fonction n'est pas disponible si vous utilisez cette version de la librairie.

[10.32.20 imagecreatefromjpeg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefromjpeg](#) (string *filename*)

[PHP 3>= 3.0.16, PHP 4 >= 4.0RC1]

[imagecreatefromjpeg\(\)](#) retourne un identifiant d'image représentant un image obtenu à partir du fichier *filename*.

[imagecreatefromjpeg\(\)](#) retourne une chaîne vide en cas d'échec. Elle affiche aussi un message d'erreur, qui s'affiche comme un lien brisé dans un navigateur web. Pour faciliter le débogage, voici une erreur **JPEG**:

Exemple de gestion d'erreur lors de la création d'image (gracieusement offert par vic@zysys.com)

```

<?php
function loadjpeg($imgname) {
    $im = @imagecreatefromjpeg($imgname); /* Tentative d'ouverture */
    if (!$im) { /* Vérification */
        $im = imagecreate(150, 30); /* Création d'une image blanche */
        $bgc = imagecolorallocate($im, 255, 255, 255);
        $tc = imagecolorallocate($im, 0, 0, 0);
        ImageFilledRectangle($im, 0, 0, 150, 30, $bgc);
        /* Affichage d'un message d'erreur */
        imagestring($im, 1, 5, 5, "Erreur de chargement de l'image $imgname", $tc);
    }
    return $im;
}
?>

```

[10.32.21 imagecreatefrompng](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefrompng](#) (string *filename*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[imagecreatefrompng\(\)](#) retourne un identifiant d'image représentant un image obtenu à partir du fichier *filename*.

[imagecreatefrompng\(\)](#) retourne une chaîne vide en cas d'échec. Elle affiche aussi un message d'erreur, qui s'affiche comme un lien brisé dans un navigateur web. Pour faciliter le débogage, voici une erreur **PNG**:

Exemple de gestion d'erreur lors de la création d'image (gracieusement offert par vic@zysys.com)

```

<?php
function LoadPNG($imgname) {
    $im = @imagecreatefrompng($imgname); /* Tentative d'ouverture */
    if (!$im) { /* Vérification */
        $im = imagecreate(150, 30); /* Création d'une image blanche */
        $bgc = imagecolorallocate($im, 255, 255, 255);
        $tc = imagecolorallocate($im, 0, 0, 0);
        imagefilledrectangle($im, 0, 0, 150, 30, $bgc);
        /* Affichage d'un message d'erreur */
        imagestring($im, 1, 5, 5, "Erreur de chargement de l'image $imgname", $tc);
    }
    return $im;
}

```

?>

[10.32.22 imagecreatefromwbmp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefromwbmp](#)(string *filename*)

[PHP 4 >= 4.0.1]

[imagecreatefromwbmp\(\)](#) retourne une ressource d'image PHP, représentant l'image *filename*.

[imagecreatefromwbmp\(\)](#) retourne une chaîne vide en cas d'erreur. Il retourne aussi un message d'erreur qui s'affiche comme un lien mort dans un navigateur. Pour aider au débogage, l'exemple suivant va produire une erreur **WBMP**:

Exemple de gestion des erreurs durant la création d'une image wbmp (gracieusement proposé par vic@zysys.com)

```

function loadwbmp($imgname) {
    $im = @imagecreatefromwbmp($imgname); /* Tentative d'ouverture */
    if (!$im) { /* Vérification que cela s'est bien passé */
        $im = imagecreate(20, 20); /* Crée une image blanche */
        $bgc = imagecolorallocate($im, 255, 255, 255);
        $tc = imagecolorallocate($im, 0, 0, 0);
        imagefilledrectangle($im, 0, 0, 10, 10, $bgc);
        /* Affiche le message d'erreur */
        imagestring($im, 1, 5, 5, "Erreur de chargement de $imgname", $tc);
    }
    return $im;
}

```

[10.32.23 imagecreatefromstring](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagecreatefromstring](#)(string *string*)

[imagecreatefromstring\(\)](#) retourne un identifiant d'image représentant la chaîne *string*.

[10.32.24 imagedashedline](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagedashedline](#)(int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[PHP 3, PHP 4]

[imagedashedline\(\)](#) dessine une ligne pointillée entre les points (*x1,y1*) et (*x2,y2*) (le coin supérieur droit est l'origine (0,0)) dans l'image *im*, avec la couleur *col*.

Voir aussi [imageline\(\)](#).

[10.32.25 imagedestroy](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagedestroy](#) (int *im*)

[PHP 3, PHP 4]

[imagedestroy\(\)](#) libère toute la mémoire associée avec l'image *im*. *im* est un identifiant d'image valide retourné par [imagecreate\(\)](#).

[10.32.26 imagefill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefill](#) (int *im*, int *x*, int *y*, int *col*)

[PHP 3, PHP 4]

[imagefill\(\)](#) effectue un remplissage avec la couleur *col*, dans l'image *im*, à partir du point de coordonnées (*x*, *y*) (le coin supérieur gauche est l'origine (0,0)).

[10.32.27 imagefilledpolygon](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefilledpolygon](#) (int *im*, array *points*, int *num_points*, int *col*)

[PHP 3, PHP 4]

[imagefilledpolygon\(\)](#) dessine un polygone rempli dans l'image *im*. *points* est un tableau PHP qui contient les sommets des polygones sous la forme : points[0] = x0, points[1] = y0, points[2] = x1, points[3] = y1, etc. *num_points* est le nombre total de sommets.

[10.32.28 imagefilledrectangle](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefilledrectangle](#) (int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[PHP 3, PHP 4]

[imagefilledrectangle\(\)](#) dessine un rectangle de couleur *col* dans l'image *im*, en commençant par le sommet supérieur gauche (*x1*, *y1*) et finissant au sommet inférieur droit (*x2*, *y2*). Le coin supérieur gauche est l'origine (0, 0).

[10.32.29 imagefilltoborder](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefilltoborder](#) (int *im*, int *x*, int *y*, int *border*, int *col*)

[PHP 3, PHP 4]

[imagefilltoborder\(\)](#) remplit avec la couleur *col* toute la région à l'intérieur de la région limitée par la couleur *border*. Le point de départ est (*x*,*y*) (le coin supérieur gauche est l'origine (0,0)).

10.32.30 imagefontheight

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefontheight](#) (int *font*)

[PHP 3, PHP 4]

[imagefontheight\(\)](#) retourne la hauteur de la police en pixel.

Voir aussi [imagefontwidth\(\)](#) et [imageloadfont\(\)](#).

10.32.31 imagefontwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagefontwidth](#) (int *font*)

[PHP 3, PHP 4]

[imagefontwidth\(\)](#) retourne la largeur de la police en pixels.

Voir aussi [imagefontheight\(\)](#) et [imageloadfont\(\)](#).

10.32.32 imagegif

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagegif](#) (int *im*, string *filename*)

[PHP 3, PHP 4]

[imagegif\(\)](#) crée un fichier image **GIF** avec le nom *filename* d'après l'image *im*. L'argument *im* est un identifiant valide retourné par la fonction [imagecreate\(\)](#).

Le format de l'image sera GIF87a à moins que l'image n'ait une couleur transparente (mise en place grâce à la fonction [imagecolortransparent\(\)](#)), ce qui fera qu'elle sera au format GIF89a.

Le nom du fichier est optionnel, et dans ce cas, l'image sera transmise directement à la sortie standard. En envoyant une image de type image/gifcontent-type, (grâce à la fonction [header\(\)](#)), vous pouvez créer des images avec des scripts PHP. Note : *Etant donné que toutes les fonctions **GIF** ont été supprimées de la bibliothèque GD version 1.6, cette fonction ne sera pas accessible si vous avez cette version de la librairie. Le code suivant vous permet d'écrire des scripts PHP plus portables : le type de GD est automatiquement détecté. Il remplace la séquence Header("Content-type: image/gif"); ImageGif(\$im); par un code plus souple :*

```
<?php
if (function_exists("imagegif")) {
header("Content-type: image/gif");
imagegif($im);

elseif (function_exists("imageJpeg")) {
header("Content-type: image/jpeg");
imagejpeg($im, "", 0.5);
}
elseif (function_exists("imagePng")) {
header("Content-type: image/png");
imagepng($im);
} elseif (function_exists("imagewbmp")) {
header("Content-type: image/vnd.wap.wbmp");
imagewbmp($im);
```

```

    } else
die("Pas de support graphique avec PHP sur ce serveur");
?>

```

} Note : En PHP 4, à partir de la version 4.0.2, vous pouvez utiliser la fonction [imagetypes\(\)](#) à la place de [function_exists\(\)](#) pour vérifier que certains formats d'images sont supportés :

```

<?php
if (function_exists("imagegif")) {
header("Content-type: image/gif");
imagegif($im);

elseif (function_exists("imageJpeg")) {
header("Content-type: image/jpeg");
imagejpeg($im, "", 0.5);
}
elseif (function_exists("imagePng")) {
header("Content-type: image/png");
imagepng($im);
} elseif (function_exists("imagewbmp")) {
header("Content-type: image/vnd.wap.wbmp");
imagewbmp($im);
} else
die("Pas de support graphique avec PHP sur ce serveur");
?>

```

}
Voir aussi [imagepng\(\)](#), [imagewbmp\(\)](#), [imagejpeg\(\)](#), [imagetypes\(\)](#).

10.32.33 imagepng

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagepng](#) (int *im*, string *filename*)
[PHP 3 >= 3.0.13, PHP 4 >= 4.0b4]

[imagepng\(\)](#) envoie l'image GD (*im*) au format **PNG** sur la sortie standard (typiquement, le navigateur web), ou si *filename* est fourni, l'envoi dans un fichier.

```

<?php
$im = imagecreatefrompng("test.png");
imagepng($im);
?>

```

Le nom du fichier est optionnel, et dans ce cas, l'image sera transmise directement à la sortie standard. En envoyant une image de type image/png content-type (grâce à la fonction [header\(\)](#)), vous pouvez créer des images **PNG** avec des scripts PHP.

Voir aussi [imagegif\(\)](#), [imagewbmp\(\)](#), [imagejpeg\(\)](#), [imagetypes\(\)](#).

10.32.34 imagejpeg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagejpeg](#) (int *im*, string *filename* , int *quality*)

[PHP 3>= 3.0.16, PHP 4 >= 4.0RC1]

[imagejpeg\(\)](#) envoie l'image GD (*im*) au format **JPEG** sur la sortie standard (typiquement, le navigateur web), ou si *filename* est fourni, l'envoi dans un fichier. *im* a été créé par [imagecreate\(\)](#).

Le nom du fichier est optionnel, et dans ce cas, l'image sera transmise directement à la sortie standard. En envoyant une image de type image/jpeg content-type (grâce à la fonction [header\(\)](#)), vous pouvez créer des images **JPEG** avec des scripts PHP.

Note : *Le support JPEG n'est disponible que si PHP est compilé avec GD-1.8 ou plus récent.*

Voir aussi [imagepng\(\)](#), [imagewbmp\(\)](#), [imagegif\(\)](#), [imagetypes\(\)](#).

10.32.35 imagewbmp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagewbmp](#) (int *im*, string *filename*)

[PHP 3>= 3.0.15, PHP 4 >= 4.0.1]

[imagewbmp\(\)](#) crée l'image **WBMP** dans le fichier *filename*, à partir de l'image *im*. Le paramètre *im* a été créé avec la fonction [imagecreate\(\)](#).

filename est optionnel, et s'il est omis, l'image sera envoyée directement au client. En plaçant l'entête *image/vnd.wap.wbmp*, dans le champs "content-type", vous pourrez afficher une image **WBMP**. Note : *Le support WBMP n'est disponible que si PHP a été compilé avec GD-1.8 ou plus récent.*

Voir aussi [imagepng\(\)](#), [imagegif\(\)](#), [imagejpeg\(\)](#), [imagetypes\(\)](#).

10.32.36 imageinterlace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageinterlace](#) (int *im*, int *interlace*)

[PHP 3, PHP 4]

[imageinterlace\(\)](#) active ou désactive le bit d'entrelacement. Si l'entrelacement est à 1, l'image *im* sera interlacée, et sinon, elle ne le sera pas.

[imageinterlace\(\)](#) retourne l'état courant d'entrelacement de l'image.

10.32.37 imageline

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageline](#) (int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[PHP 3, PHP 4]

[imageline\(\)](#) dessine une ligne depuis le point (*x1,y1*) jusqu'au point (*x2,y2*) (le coin supérieur gauche est l'origine (0,0)) dans l'image *im* et avec la couleur *col*.

Voir aussi [imagecreate\(\)](#) et [imagecolorallocate\(\)](#).

10.32.38 imageloadfont

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imageloadfont](#)(string *file*)

[PHP 3, PHP 4]

[imageloadfont\(\)](#) charge une nouvelle police utilisateur et retourne un identifiant sur cette police. Cet identifiant sera toujours supérieur à 5, pour éviter les conflits avec les polices standard PHP). Le format des polices dépend actuellement du système d'exploitation. Ce qui signifie qu'il vous faut générer des fichiers de polices pour la machine qui fait tourner PHP.

position	Type de donnés C	description
Octets 0–3	int	Nombre de caractères de la police
Octets 4–7	int	Valeur du premier caractère de la police (souvent 32 pour espace) @tab
Octets 8–11	int	Largeur en pixel des caractères
Octets 12–15	int	Hauteur en pixel des caractères
Octets 16–	char	Tableau avec les données des caractères, un octet par pixel pour chaque caractère, avec un total de (nombre_caractères*largeur*hauteur) octets. @tab

Voir aussi [imagefontwidth\(\)](#) et [imagefontheight\(\)](#).

10.32.39 imagepolygon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagepolygon](#)(int *im*, array *points*, int *num_points*, int *col*)

[PHP 3, PHP 4]

[imagepolygon\(\)](#) dessine un polygone dans l'image *im*. *points* est un tableau PHP qui contient les sommets du polygone sous la forme : points[0] = x0, points[1] = y0, points[2] = x1, points[3] = y1, etc. num_points est le nombre de sommets.

Voir aussi [imagecreate\(\)](#).

10.32.40 imagepsbbox

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagepsbbox](#)(string *text*, int *font*, int *size*, int *space*, int *width*, float *angle*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

size est exprimé en pixels.

space permet de changer la valeur par défaut du caractère espace. Cette valeur est ajoutée lors des dessins, et donc, peut être négative.

tightness permet de contrôler la quantité d'espace entre les caractères. Cette quantité est ajouté lors des

dessins, et peut donc être négative.

angle est en degrés.

Les paramètres ***space*** et ***tightness*** sont exprimés en unité d'espacement de caractères, avec 1 unité vaut 1/1000 d'un em carré (NDT : kesako?).

Les paramètres ***space***, ***tightness*** et ***angle*** sont optionnels.

Le rectangle entourant est calculé en utilisant les informations disponibles sur les tailles de caractères, et, malheureusement, ont tendance à être légèrement différents du résultat réel final. Si l'angle est de 0 degré, vous pouvez vous attendre à avoir besoin d'un rectangle d'au moins un pixel plus grand dans toutes les directions.

[`imagepsbbox\(\)`](#) retourne un tableau contenant les éléments suivants :

0	Abscisse inférieure gauche
1	Ordonnée inférieure gauche
2	Abscisse supérieure droite
3	Ordonnée supérieure droite

Voir aussi [`imagepstext\(\)`](#).

[10.32.41 imagepsencodefont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`imagepsencodefont`](#)(int *font_index*, string *encodingfile*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

[`imagepsencodefont\(\)`](#) charge le codage vectoriel d'un caractère depuis un fichier et change le codage vectoriel de la police correspondante. Etant donné que les polices PostScript ne disposent pas des caractères au-delà de 127, vous aurez sûrement besoin de les changer sur vous utilisez une autre langue que l'anglais. Le format exact est décrit dans la documentation T1libs. T1lib est disponible en deux formes : IsoLatin1.enc et IsoLatin2.enc.

Si vous commencez à utiliser cette fonction régulièrement, une meilleure solution est de définir un encodage, et de l'utiliser avec `set ps.default_encoding` dans [7.1 Le fichier de configuration](#) pour utiliser par défaut l'encodage correct.

[10.32.42 imagepsfreefont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`imagepsfreefont`](#)(int *fontindex*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

Voir aussi [`imagepsloadfont\(\)`](#).

[10.32.43 imagepsloadfont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`imagepsloadfont`](#)(string *filename*)

[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

Au cas où tout a bien marché, un index de police va être retourné, et pourra être utilisé pour des opérations ultérieures. Sinon, la fonction retourne FALSE et affiche un message décrivant ce qui est erroné.

Voir aussi [imagepsfreefont\(\)](#).

[10.32.44 imagepsextendfont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [imagepsextendfont](#) (int *font_index* , double *extend*)
[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

Etend ou condense la police de caractères *font_index*. Si la valeur de *extend* est inférieure à 1, ce sera une condensation.

[10.32.45 imagepslantfont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [imagepslantfont](#) (int *font_index* , double *slant*)
[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

Met en italique la police de caractères *font_index* avec le coefficient *slant*.

[10.32.46 imagepstext](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagepstext](#) (int *image*, string *text*, int *font*, int *size*, int *foreground*, int *background*, int *x*, int *y*, int *space*, int *tightness*, float *angle*, int *antialias_steps*)
[PHP 3>= 3.0.9, PHP 4 >= 4.0RC1]

size est exprimé en pixels.

foreground est la couleur dans laquelle le texte va être dessiné. *background* est la couleur d'anti aliasing. Aucun pixel avec la couleur *background* n'est dessiné, ce qui fait que l'arrière plan n'a pas besoin d'être dans une couleur fixe.

Les coordonnées données (*x*, *y*) définissent l'origine du premier caractère (grossièrement, le coin inférieur gauche du caractère). Ceci est différent de la fonction [imagestring\(\)](#), où (*x*, *y*) définissait le coin supérieur gauche du premier caractère. Reportez vous à la documentation PostScript pour avoir des détails à propos des polices et de leurs tailles.

space permet de changer la taille par défaut du caractère d'espacement. Cette valeur peut être négative.

tightness permet de contrôler la quantité d'espace entre deux caractères. Cette valeur peut être négative.

angle est en degrés.

antialias_steps permet de contrôler le nombre de couleurs du texte anti-aliasé. Les valeurs autorisées sont 4 et 16. 16 est recommandé pour les polices de moins de 20 pixels, car l'effet est alors visible. Avec les tailles plus grandes, utilisez de préférence 4, qui est moins gourmande en ressources.

Les paramètres *space* et *tightness* sont exprimés en unité d'espace caractère, ce qui vaut 1/1000ème d'un em-carré (? ? ?).

Les paramètres *space*, *tightness*, *angle* et *antialias* sont optionnels.

[imagepstext\(\)](#) retourne un tableau contenant les éléments suivants :

0	Abscisse inférieure gauche
---	----------------------------

1	Ordonnée inférieure gauche
2	Abscisse supérieure droite
3	Ordonnée supérieure droite

Voir aussi [imagepsbbox\(\)](#).

10.32.47 imagerectangle

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagerectangle](#) (int *im*, int *x1*, int *y1*, int *x2*, int *y2*, int *col*)

[PHP 3, PHP 4]

[imagerectangle\(\)](#) dessine un rectangle dans la couleur *col*, dans l'image *im*, et en commençant au point supérieur gauche (*x1*, *y1*), et en finissant au point inférieur droit (*x2*, *y2*). Le coin supérieur gauche est l'origine (0,0).

10.32.48 imagesetpixel

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagesetpixel](#) (int *im*, int *x*, int *y*, int *col*)

[PHP 3, PHP 4]

[imagesetpixel\(\)](#) dessine un pixel au point (*x*,*y*) (le coin supérieur gauche est l'origine (0,0)) dans l'image *im*, et avec la couleur *col*.

Voir aussi [imagecreate\(\)](#) et [imagecolorallocate\(\)](#).

10.32.49 imagestring

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagestring](#) (int *im*, int *font*, int *x*, int *y*, string *s*, int *col*)

[PHP 3, PHP 4]

[imagestring\(\)](#) dessine une chaîne sur une ligne horizontale, dans l'image *im*, aux coordonnées (*x*,*y*) (le coin supérieur gauche est l'origine (0,0)) dans la couleur *col*. Si l'argument de police vaut 1, 2, 3, 4 ou 5, une des polices par défaut sera utilisée).

Voir aussi [imageloadfont\(\)](#).

10.32.50 imagestringup

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagestringup](#) (int *im*, int *font*, int *x*, int *y*, string *s*, int *col*)

[PHP 3, PHP 4]

[imagestringup\(\)](#) dessine une chaîne sur une ligne verticale dans l'image *im* aux coordonnées (*x*, *y*) (l'origine est le coin supérieur gauche (0,0)) dans la couleur *col*. Si la police utilisée est 1, 2, 3, 4 ou 5, une police par

défaut sera utilisée.

Voir aussi [imagerloadfont\(\)](#).

10.32.51 imagesx

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagesx](#) (int *im*)

[PHP 3, PHP 4]

[imagesx\(\)](#) retourne la largeur de l'image référencée par *im*.

Voir aussi [imagecreate\(\)](#) et [imagesy\(\)](#).

10.32.52 imagesy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagesy](#) (int *im*)

[PHP 3, PHP 4]

[imagesy\(\)](#) retourne la hauteur de l'image référencée par *im*.

Voir aussi [imagecreate\(\)](#) et [imagesx\(\)](#).

10.32.53 imagettfbbox

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagettfbbox](#) (int *size*, int *angle*, string *fontfile*, string *text*)

[PHP 3>= 3.0.1, PHP 4]

[imagettfbbox\(\)](#) calcule et retourne le rectangle entourant le texte *text*, écrit avec une police truetype.

text

- La chaîne à mesurer.

size

- La taille de la police en pixel.

fontfile

- Le nom de la police TrueType (peut aussi être une URL.)

angle

- Angle en degré dans lequel le texte *text* va être mesuré.

[imagettfbbox\(\)](#) retourne un tableau avec 8 éléments, représentant les 4 sommets du rectangle ainsi définis.

0	Coin inférieur gauche, abscisse
1	Coin inférieur gauche, ordonnée
2	Coin inférieur droit, abscisse
3	Coin inférieur droit, ordonnée
4	Coin supérieur droit, abscisse

5	Coin supérieur droit, ordonnée
6	Coin supérieur gauche, abscisse
7	Coin supérieur gauche, ordonnée

Les positions des points sont relatives au texte *text*, indépendamment de l'angle : coin supérieur gauche faire référence au coin supérieur gauche du texte écrit horizontalement.

[imageftbbox\(\)](#) requiert les bibliothèques GD et FreeType.

Voir aussi [imagefttext\(\)](#).

10.32.54 imagefttext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imagefttext](#) (int *im*, int *size*, int *angle*, int *x*, int *y*, int *col*, string *fontfile*, string *text*)

[PHP 3, PHP 4]

[imagefttext\(\)](#) dessine la chaîne *text* dans l'image *im*, en commençant aux coordonnées (*x,y*) (le coin supérieur gauche est l'origine (0,0)), avec un angle de *angle*, et dans la couleur *col*, en utilisant la police TrueType identifiée par *fontfile*.

Les coordonnées (*x,y*) serviront de référence pour le premier caractère (en gros, le coin inférieur gauche du caractère). C'est différent de [imagestring\(\)](#), qui utilise le coin supérieur droit.

angle est donné en degrés, avec degré 0 pour un texte horizontal, et en comptant les angles dans le sens inverse des aiguilles d'une montre (sens direct).

fontfile est le chemin jusqu'à la police TrueType à utiliser.

text est le texte à dessiner, incluant aussi des séquences de caractères UTF-8 (de la forme: {) pour générer des caractères au delà de 255.

col est l'index de la couleur dans la palette. Utiliser des index négatifs, revient à supprimer l'anti-aliasing.

[imagefttext\(\)](#) retourne un tableau de 8 éléments représentant les 4 points marquant les limites du texte.

L'ordre des points est : supérieur gauche, supérieur droit, inférieur droit, inférieur gauche. Les points sont nommés relativement au texte à l'horizontal.

Cet exemple va générer une image **GIF** noire de 400x30 pixels, avec les mots "Test en cours..." en police blanche, Arial.

imagefttext()

```
<?php
Header("Content-type: image/gif");
$im = imagecreate(400,30);
$black = ImageColorAllocate($im, 0,0,0);
$white = ImageColorAllocate($im, 255,255,255);
ImageTTFText($im, 20, 0, 10, 20, $white, "/path/arial.ttf", "Test en cours... Omega: &#937;");
ImageGif($im);
ImageDestroy($im);
?>
```

[imagefttext\(\)](#) requiert les bibliothèques GD et [FreeType](#).

Voir aussi [imageftbbox\(\)](#).

10.32.55 imagetypes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imagetypes](#)

[PHP 4 >= 4.0.2]

[imagetypes\(\)](#) retourne un champs de bit correspondant aux formats d'images supportés par la version de GD utilisée. Les valeurs suivantes sont valables : IMG_GIF | IMG_JPG | IMG_PNG | IMG_WBMP. Pour vous assurer du support *PNG*, faites ceci :

Exemple avec ImageTypes

```
<?php
if (imagetypes() & IMG_PNG) {
echo "Le type PNG est supporté";
}
?>
```

10.32.56 read_exif_data

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [read_exif_data](#)(string *filename*)

[PHP 4 >= 4.0.1]

[read_exif_data\(\)](#) lit les entêtes EXIF de l'image *JPEG filename*. Elle retourne un tableau associatif où les index sont les noms d'entêtes EXIF, et les valeurs sont leur valeur associée. Les entêtes EXIF sont souvent disponibles dans les images générées par les appareils photos digitaux, mais chaque constructeur marque ses images d'une manière qui lui est propre : il est impossible de savoir quels entêtes seront présents.

read_exif_data

```
<?php
$exif = read_exif_data('p0001807.jpg');
while(list($k,$v)=each($exif)) {
echo "$k: $v<br>\n";
}
?>
```

Cet exemple va afficher : FileName: p0001807.jpg FileDateTime: 929353056
FileSize: 378599 CameraMake: Eastman Kodak Company CameraModel: KODAK
DC265 ZOOM DIGITAL CAMERA (V01.00) DateTime: 1999:06:14 01:37:36 Height:
1024 Width: 1536 IsColor: 1 FlashUsed: 0 FocalLength: 8.0mm
RawFocalLength: 8 ExposureTime: 0.004 s (1/250) RawExposureTime:
0.0040000001899898 ApertureFNumber: f/ 9.5 RawApertureFNumber:
9.5100002288818 FocusDistance: 16.66m RawFocusDistance: 16.659999847412
Orientation: 1 ExifVersion: 0200

Note : [read_exif_data\(\)](#) n'est disponible que sous PHP 4 , compilé avec `--enable-exif`.

[read_exif_data\(\)](#) ne requiert par la librairie GD.

10.33 IMAP

[\[Notes en ligne\]](#)

Pour avoir accès à ces fonctions, vous devez compiler PHP avec l'option `--with-imap`. Il faut avoir installé la librairie C-client. Chargez sa dernière version sur le serveur <ftp://ftp.cac.washington.edu/imap/> et compilez-la. Puis, copiez le fichier ``c-client/c-client.a'` dans ``usr/local/lib'` ou n'importe quel autre dossier qui soit dans le chemin de link. Enfin, copiez les fichiers ``c-client/rfc822.h'`, ``mail.h'` et ``linkage.h'` dans ``usr/local/include'` ou n'importe quel autre dossier qui soit dans le chemin d'inclusion.

Ces fonctions ne sont pas limitées au protocole **IMAP**, malgré leur nom. La librairie sur laquelle elles sont développées supporte aussi **NNTP**, **POP3** et les méthodes d'accès aux boîtes aux lettres locales. Reportez-vous à la fonction [imap_open\(\)](#) pour plus d'informations.

Ce document ne peut entrer dans les détails de toutes les sujets abordés. Plus d'informations sont disponibles avec la documentation de la librairie C (``docs/internal.txt'`), ainsi que les RFC suivantes :

- [RFC821](#): Simple Mail Transfer Protocol (**SMTP**).
- [RFC822](#): Standard for ARPA internet text messages.
- [RFC2060](#): Internet Message Access Protocol (**IMAP**) Version 4rev1.
- [RFC1939](#): Post Office Protocol Version 3 (**POP3**).
- [RFC977](#): Network News Transfer Protocol (**NNTP**).
- [RFC2076](#): Common Internet Message Headers.
- [RFC2045](#), [RFC2046](#), [RFC2047](#), [RFC2048](#) & [RFC2049](#): Multipurpose Internet Mail Extensions (MIME).

Une étude approfondie est aussi disponibles dans les livres suivants (en anglais): [Programming Internet Email](#) par David Wood et [Managing IMAP](#) par Dianna Mullet & Kevin Mullet.

10.33.1 imap_append

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`int imap_append(int imap_stream, string mbbox, string message, string flags)`
[PHP 3, PHP 4]

[imap_append\(\)](#) ajoute un message dans la boîte aux lettres *mbbox*. Si l'option *flags* est utilisée, *flags* sera aussi écrit dans la boîte aux lettres.

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Lors des échanges avec le serveur Cyrus IMAP, vous devrez utiliser `"\r\n"` comme terminaison de ligne, à la place de `"\n"` ou l'opération échouera.

Exemple avec `imap_append()`

```
<?php
$stream = imap_open("{your.imap.host}INBOX.Drafts","username", "password");
$check = imap_check($stream);
print "Nombre de message avant ajout : ". $check->Nmsgs."\n";
imap_append($stream, "{your.imap.host}INBOX.Drafts"
    , "From: me@my.host\r\n"
    . "To: you@your.host\r\n"
    . "Subject: test\r\n"
    . "\r\n"
    . "Ceci est un message de test. Ignorez le\r\n"
```

```

    );
    $check = imap_check($stream);
    print "Nombre de message après ajout : ". $check->Nmsgs."\n";
    imap_close($stream);
?>

```

10.33.2 imap_base64

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_base64](#)(string *text*)
[PHP 3, PHP 4]

[imap_base64\(\)](#) décode un texte encodé en BASE64. Le texte décodé est retourné sous la forme d'une chaîne.

10.33.3 imap_body

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_body](#)(int *imap_stream*, int *msg_number*, int *flags*)
[PHP 3, PHP 4]

[imap_body\(\)](#) retourne le corps du message numéro *msg_number* de la boîte aux lettres courante. L'option *flags* est un masque qui peut contenir les valeurs suivantes :

- FT_UID – msgno est un UID
- FT_PEEK – Ne pas lever le drapeaux \Seen (Message lu) si il n'est pas déjà levé.
- FT_INTERNAL – La chaîne renvoyée est au format interne, et ne va pas canoniser les CRLF.

[imap_body\(\)](#) va retourner une copie brute du corps du message. Pour extraire les sous parties MIME du message, utilisez [imap_fetchstructure\(\)](#) pour analyser la structure, et [imap_fetchbody\(\)](#) pour extraire une copie d'une des sous-partie.

10.33.4 imap_check

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_check](#)(int *imap_stream*)
[PHP 3, PHP 4]

[imap_check\(\)](#) retourne les informations à propos de la boîte aux lettres courante. Retourne FALSE en cas d'échec.

[imap_check\(\)](#) vérifie le statut de la boîte aux lettres courante, sur le serveur *imap_stream*, et retourne les informations dans un objet avec les membres suivants :

- Date – Date de dernière modification du contenu de la boîte aux lettres
- Driver – protocole utilisé pour accéder à la boîte aux lettres: **POP3**, **IMAP**, **NNTP**.
- Mailbox – nom de la boîte aux lettres
- Nmsgs – nombre de messages de la boîte aux lettres

- Recent – nombre de messages récents de la boîte aux lettres

10.33.5 [imap_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_close](#)(int *imap_stream*, int *flags*)

[PHP 3, PHP 4]

[imap_close\(\)](#) termine un flot **IMAP**. [imap_close\(\)](#) prend un argument optionnel *flag*, CL_EXPUNGE, qui va retirer automatiquement de la liste la boîte aux lettres.

10.33.6 [imap_createmailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_createmailbox](#)(int *imap_stream*, string *mbox*)

[PHP 3, PHP 4]

[imap_createmailbox\(\)](#) crée une nouvelle boîte aux lettres nommée *mbox*. Les noms contenant des caractères spéciaux doivent être encodés.

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Exemple avec imap_createmailbox()

```
<?php
$mbox = imap_open("{your.imap.host}", "utilisateur", "mot_de_passe", OP_HALFOPEN)
    || die("connexion impossible: ".imap_last_error());
$name1 = "nouvellepbox";
$name2 = imap_utf7_encode("nouvellepboxéx");
$newname = $name1;
echo "Le nouveau nom sera '$name1'<br>\n";
# Nous allons créer maintenant une nouvelle boîte aux lettres "phptestbox"
# dans votre dossier inbox, vérifier son état et finalement, la supprimer
# pour remettre votre inbox dans son état initial.
if(@imap_createmailbox($mbox,imap_utf7_encode("{your.imap.host}INBOX.$newname")) {
    $status = @imap_status($mbox, "{your.imap.host}INBOX.$newname", SA_ALL);
    if($status) {
        print("Votre nouvelle boîte '$name1' est dans l'état suivant :<br>\n");
        print("Messages:      ". $status->messages      )."<br>\n";
        print("Récent:        ". $status->recent         )."<br>\n";
        print("Non lus:         ". $status->unseen          )."<br>\n";
        print("UID suivant:     ". $status->uidnext         )."<br>\n";
        print("UID validité:    ". $status->uidvalidity      )."<br>\n";
        if(imap_renamemailbox($mbox,"{your.imap.host}INBOX.$newname","{your.imap.host}INBOX.$name2")) {
            echo "renommage de la boîte aux lettres '$name1' en '$name2'<br>\n";
            $newname=$name2;
        } else {
            print "imap_renamemailbox sur la nouvelle boîte aux lettres a échoué : ".imap_last_error()."<br>\n";
        }
    } else {
        print "imap_status sur la nouvelle boîte aux lettres a échoué : ".imap_last_error()."<br>\n";
    }
}
if(@imap_deletemailbox($mbox,"{your.imap.host}INBOX.$newname")) {
    print "new mailbox supprimée pour remettre tout en état<br>\n";
}
```



```

    } else {
print "imap_deletemailbox ur la nouvelle boîte aux lettres a échoué : ".implode("<br>\n",imap_er
    }
} else {
print "Impossible de créer une nouvelle boîte aux lettres : ".implode("<br>\n",imap_errors())."<
    }
imap_close($mbox);
?>

```

Voir aussi [imap_renamemailbox\(\)](#), [imap_deletemailbox\(\)](#) et [imap_open\(\)](#) pour connaître le format des noms de *mbox*.

10.33.7 imap_delete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_delete](#) (int *imap_stream*, int *msg_number*, int *flags*)
[PHP 3, PHP 4]

[imap_delete\(\)](#) retourne TRUE.

[imap_delete\(\)](#) marque le fichier *msg_number* pour l'effacement, dans la boîte aux lettres courante. Le paramètre optionnel *flags* ne prend qu'une seule valeur, *FT_UID*, qui indique à PHP qu'il faut traiter *msg_number* comme un *UID*. L'effacement réel n'interviendra que lors de l'appel de la fonction [imap_expunge\(\)](#).

Exemple imap_delete()

```

<?php
$mbox = imap_open ("{your.imap.host}INBOX", "utilisateur", "mot_de_passe")
|| die ("connexion impossible: " . imap_last_error());
$check = imap_mailboxmsginfo ($mbox);
print "Nombre de messages avant effacement  : " . $check->Nmsgs . "<br>\n" ;
imap_delete ($mbox, 1);
$check = imap_mailboxmsginfo ($mbox);
print "Nombre de messages après effacement: " . $check->Nmsgs . "<br>\n" ;
imap_expunge ($mbox);
$check = imap_mailboxmsginfo ($mbox);
print "Nombre de messages après imap_expunge: " . $check->Nmsgs . "<br>\n" ;
imap_close ($mbox);
?>

```

10.33.8 imap_deletemailbox

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_deletemailbox](#) (int *imap_stream*, string *mbox*)
[PHP 3, PHP 4]

[imap_deletemailbox\(\)](#) efface la boîte aux lettres (voir [imap_open\(\)](#) pour connaître le format des noms de *mbox*).

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Voir aussi [imap_createmailbox\(\)](#), [imap_renamemailbox\(\)](#), et [imap_open\(\)](#) pour le format du paramètre *mbox*.

[10.33.9 imap_expunge](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_expunge](#)(int *imap_stream*)

[PHP 3, PHP 4]

[imap_expunge\(\)](#) efface tous les messages marqués pour l'effacement par [imap_delete\(\)](#).

Retourne TRUE.

[10.33.10 imap_fetchbody](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_fetchbody](#)(int *imap_stream*, int *msg_number*, string *part_number*, flags)

[PHP 3, PHP 4]

[imap_fetchbody\(\)](#) va rechercher une section du corps du message, et la retourne sous la forme d'une chaîne.

La section est une chaîne d'entiers, séparés par des virgules, qui servent d'index dans le corps du message, comme spécifié dans la norme IMAP4. Le texte n'est alors pas décodé par [imap_fetchbody\(\)](#).

L'option [imap_fetchbody\(\)](#) est un masque qui peut contenir les valeurs suivantes :

- FT_UID – msgono est un UID
- FT_PEEK – Ne pas lever le drapeau \Seen (Message lu) si il n'est pas déjà levé.
- FT_INTERNAL – La chaîne renvoyée est au format interne, et ne va pas canoniser les CRLF.

[10.33.11 imap_fetchstructure](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_fetchstructure](#)(int *imap_stream*, int *msg_number*, int *flags*)

[PHP 3, PHP 4]

[imap_fetchstructure\(\)](#) la structure du message *msg_number*. [imap_fetchstructure\(\)](#) dispose d'une option *flags*, qui une seule valeur, *FT_UID*, pour indiquer que l'argument *msg_number* est un *UID*. Cette fonction retourne un objet avec des propriétés d'enveloppe, de date interne, de taille, de structure de flags et de corps, ainsi qu'un objet pour chaque attachement. La structure est la suivante :

type	Type primaire de corps
encoding	Codage de transfert du corps
ifsubtype	TRUE si il y a une chaîne de sous type
subtype	sous type <i>MIME</i>
ifdescription	TRUE si il y au ne chaîne de description
description	Chaîne de description du contenu
ifid	TRUE si il y a une chaîne d'identification

id	Chaîne d'identification
lines	Nombre de lignes
bytes	Nombre d'octets
ifdisposition	TRUE si il y a une chaîne de disposition
disposition	Chaîne de disposition
ifdparameters	TRUE s'il y a un tableau de paramètres dparameters
dparameters	tableau de disposition
ifparameters	TRUE si le tableau de paramètres existe
parameters	Tableau de paramètres <i>MIME</i>
parts	Tableau d'objet décrivant chaque partie du message

Note :

1. *dparameters* est un tableau d'objet où chaque objet à un "attribut" et une "valeur".
2. *parameter* est un tableau d'objet où chaque objet à un "attribut" et une "valeur".
3. *parts* est un tableau d'objets de même structure que l'objet supérieur, mais qui ne contient pas d'autres objets de même sorte.

0	text
1	multipart
2	message
3	application
4	audio
5	image
6	vidéo
7	autre

0	7BIT
1	8BIT
2	BINARY
3	BASE64
4	QUOTED-PRINTABLE
5	OTHER

10.33.12 [imap_headerinfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_headerinfo](#) (int *imap_stream*, int *msg_number*, int *fromlength* , int *subjectlength* , string *defaulthost*)

[PHP 3, PHP 4]

[imap_headerinfo\(\)](#) retourne un objet contenant divers éléments d'entête.

remail, date, Date, subject, Subject, in_reply_to, message_id,
newsgroups, followup_to, references
éléments d'entêtes:

```
Recent - 'R' si récent et lu
          'N' si récent et non lu
          ' ' si non récent
Unseen - 'U' si non lu ET non récent
          ' ' si lu OU non lu et récent
Answered - 'A' si répondu,
           ' ' si non répondu
Deleted - 'D' si effacé,
          ' ' si non effacé
Draft - 'X' si brouillon,
        ' ' si non brouillon
Flagged - 'F' si marqué,
          ' ' si non marqué
```

Notez bien que le comportement récent/non lu est un peu particulier :
si vous voulez savoir si un message est non lu, vous devez le vérifier
avec

```
Unseen == 'U' || Recent == 'N'
```

toaddress (toute la ligne d'entête To: jusqu'à 1024 caractères)

to[] (retourne un objet avec tout l'entête To, contenant):

personal

adl

mailbox

host

fromaddress (toute la ligne d'entête from: jusqu'à 1024 caractères)

from[] (retourne un objet avec tout l'entête From, contenant):

personal

adl

mailbox

host

ccaddress (toute la ligne d'entête CC: jusqu'à 1024 caractères)

cc[] (retourne un objet avec tout l'entête CC, contenant):

personal

adl

mailbox

host

bccaddress (toute la ligne d'entête BCC: jusqu'à 1024 caractères)

bcc[] (retourne un objet avec tout l'entête BCC, contenant):

personal

adl

mailbox

host

reply_toaddress (oute la ligne d'entête Reply_to: jusqu'à 1024 caractères)

reply_to[] (retourne un objet avec tout l'entête Reply_to, contenant)

personal

adl

mailbox

host

senderaddress (toute la ligne d'entête Sender: jusqu'à 1024 caractères)
 sender[] (retourne un objet avec tout l'entête Sender, contenant)
 personal
 adl
 mailbox
 host
 return_path (toute la ligne d'entête Return-path: jusqu'à 1024 caractères)
 return_path[] (retourne un objet avec tout l'entête Return-path, contenant)
 personal
 adl
 mailbox
 host
 udate (Date du mail, au format UNIX)
 fetchfrom (Ligne d'entête from formatée pour tenir dans **fromlength** caractères)
 fetchsubject (Ligne d'entête subject formatée pour tenir dans **subjectlength** caractères)

10.33.13 imap_header

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_header](#) (int *imap_stream*, int *msg_number*, int *fromlength* , int *subjectlength* , string *defaulthost*)

[PHP 3, PHP 4]

Cette fonction est un alias de [imap_headerinfo\(\)](#) et lui est identique en tout point.

10.33.14 imap_rfc822_parse_headers

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_rfc822_parse_headers](#) (string *headers*, string *defaulthost*)

[PHP 4 >= 4.0RC1]

[imap_rfc822_parse_headers\(\)](#) analyse la chaîne *headers*, et retourne un objet contenant différents éléments, similaires à la fonction [imap_header\(\)](#), hormis les flags, et autres éléments liés au serveur **IMAP**.

10.33.15 imap_headers

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_headers](#) (int *imap_stream*)

[PHP 3, PHP 4]

[imap_headers\(\)](#) retourne un tableau de chaîne contenant les entête des messages. Une chaîne par message.

10.33.16 imap_listmailbox

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_listmailbox](#) (int *imap_stream*, string *ref*, string *pat*)

[PHP 3, PHP 4]

[imap_listmailbox\(\)](#) retourne un tableau contenant les noms des boîtes aux lettres.

Exemple avec `imap_listmailbox()`

```
<?php
$mbox = imap_open("{your.imap.host}", "utilisateur", "mot_de_passe", OP_HALFOPEN)
|| die("connexion impossible: ".imap_last_error());
$list = imap_listmailbox($mbox, "{your.imap.host}", "*");
if(is_array($list)) {
    reset($list);
    while (list($key, $val) = each($list))
        print imap_utf7_decode($val). "<br>\n";
    } else
    print "imap_listmailbox a échoué: ".imap_last_error(). "\n";
imap_close($mbox);
?>
```

10.33.17 [imap_getmailboxes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_getmailboxes](#) (int *imap_stream*, string *ref*, string *pat*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b4]

[imap_getmailboxes\(\)](#) retourne un tableau d'objets contenant les informations sur les boîtes aux lettres.

Chaque objet a les attributs de **name**, qui contient le nom complet de la boîte aux lettres; **delimiter**, qui est le délimiteur hiérarchique; et **attributes**. **Attributes** est un masque de bits, qui contient :

- LATT_NOINFERIORS – Cette boîte aux lettres n'a pas d'"enfants" (il n'y a plus de boîtes aux lettres en dessous de celle-ci).
- LATT_NOSELECT – Ceci est juste un container, pas une boîte aux lettres (vous ne pouvez pas l'ouvrir).
- LATT_MARKED – Cette boîte aux lettres est marquée. Utilisé uniquement avec UW-IMAPD.
- LATT_UNMARKED – Cette boîte aux lettres n'est pas marquée. Utilisé uniquement avec UW-IMAPD.

ref ne devrait être que le serveur **IMAP** sous la forme {imap_server:imap_port}, et *pattern* spécifie la position dans la hiérarchie des boîtes aux lettres, où il faut commencer à chercher. Si vous voulez passer en revue toute la hiérarchie, passez '*' comme *pattern*.

Il y a deux caractères spéciaux que vous pouvez utiliser dans *pattern* : '*' et '%'. '*' signifie : toutes les boîtes aux lettres. Si vous passez *pattern* comme '*', vous obtiendrez la liste complète des boîtes aux lettres de la hiérarchie. '%' signifie qu'on ne s'intéresse qu'au niveau courant. '%' passé à *pattern* ne retournera que les boîtes aux lettres de niveau supérieur; '~/mail/%'. Sous UW-IMAPD retournera toutes les boîtes aux lettres du dossier '~/mail directory', mais pas leurs enfants.

Exemple avec `imap_getmailboxes()`

```
<?php
$mbox = imap_open("{your.imap.host}", "utilisateur", "mot_de_passe", OP_HALFOPEN)
|| die("connexion impossible: ".imap_last_error());
$list = imap_getmailboxes($mbox, "{your.imap.host}", "*");
```

```

if(is_array($list)) {
    reset($list);
    while (list($key, $val) = each($list))
    {
        print "($key) ";
        print imap_utf7_decode($val->name).", ";
        print "'".$val->delimiter."', ";
        print $val->attributes."<br>\n";
    }
} else
    print "imap_getmailboxes a échoué : ".imap_last_error()."\n";
imap_close($mbox);
?>

```

Voir aussi [imap_getsubscribed\(\)](#).

[10.33.18 imap_listsubscribed](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_listsubscribed](#) (int *imap_stream*, string *ref*, string *pattern*)
[PHP 3, PHP 4]

[imap_listsubscribed\(\)](#) retourne un tableau avec toutes les boîtes aux lettres auxquelles vous avez souscrit. Les arguments *ref* et *pattern* indiquent respectivement, le dossier où chercher et le nom des boîtes recherchées, sous la forme d'un masque.

[10.33.19 imap_getsubscribed](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_getsubscribed](#) (int *imap_stream*, string *ref*, string *pattern*)
[PHP 3 >= 3.0.12, PHP 4 >= 4.0b4]

[imap_getsubscribed\(\)](#) est identique à [imap_getmailboxes\(\)](#), mais ne retourne que les boîtes aux lettres auxquelles l'utilisateur est inscrit.

[10.33.20 imap_mail_copy](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_mail_copy](#) (int *imap_stream*, string *msglist*, string *mbox*, int *flags*)
[PHP 3, PHP 4]

[imap_mail_copy\(\)](#) copie les messages email spécifiés par *msglist* dans la boîte aux lettres nommée *mbox*. *msglist* est un intervalle, et pas seulement une liste numéros de message.

[imap_mail_copy\(\)](#) retourne TRUE en cas de succès et FALSE en cas d'erreur.

flags est un masque, qui peut contenir une ou plusieurs des valeurs suivantes :

- CP_UID – la séquence de nombre contient des UIDS
- CP_MOVE – Efface les messages après copie.

[10.33.21 imap_mail_move](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_mail_move](#)(int *imap_stream*, string *msglist*, string *mbbox*, int *flags*)

[PHP 3, PHP 4]

[imap_mail_move\(\)](#) déplace les messages spécifiés par *msglist* dans la boîte aux lettres *mbbox*. *msglist* est un intervalle, et pas seulement une liste de messages.

flags est un champs de bit et peut contenir une seule valeur :

- CP_UID – La séquence de nombrs contient UIDS

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[10.33.22 imap_num_msg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_num_msg](#)(int *imap_stream*)

[PHP 3, PHP 4]

[imap_num_msg\(\)](#) retourne le nombre de message dans la boîte aux lettres courante.

[10.33.23 imap_num_recent](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_num_recent](#)(int *imap_stream*)

[PHP 3, PHP 4]

[imap_num_recent\(\)](#) retourne le nombre de message récents dans la boîte aux lettres courante.

[10.33.24 imap_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_open](#)(string *mailbox*, string *username*, string *password*, int *flags*)

[PHP 3, PHP 4]

[imap_open\(\)](#) retourne un flot **IMAP** en cas de succès, et FALSE en cas d'erreur. [imap_open\(\)](#) peut aussi être utilisée pour ouvrir des flots sur des serveurs **POP3** et **NNTP**.

Un nom de boîte aux lettres est constitué d'une adresse de serveur, et d'une adresse de boîte sur ce serveur. Le mot réservé INBOX représente la boîte aux lettres de l'utilisateur courant. L'adresse du serveur, mise entre accolades '{' et '}', est constitué du nom du serveur ou de son adresse IP, d'une spécification de protocole (commençant par '/') et d'un port optionnel (spécifié avec ':'). Cette partie est obligatoire dans les paramètres de la boîte aux lettres. Les noms de boîtes aux lettres qui contiennent des caractères spéciaux (en dehors de l'espace ASCII) doivent être encodés avec [imap_utf7_encode\(\)](#).

Les options sont un masque de bit, qui peut prendre une ou plusieurs des valeurs suivantes :

- **OP_READONLY** – Ouvre une boîte aux lettres en lecture seule
- **OP_ANONYMOUS** – Ne pas utiliser, ou modifier le fichier `.newsrsrc` pour les news.
- **OP_HALFOPEN** – Pour les noms **IMAP** et **NNTP**, ouvre une connexion mais n'ouvre pas une boîte aux lettres.
- **CL_EXPUNGE** – Supprime automatiquement la boîte aux lettres de la liste, lors de la terminaison du flot.

Pour se connecter à un serveur **IMAP**, on peut utiliser la commande suivante :

```
<?php
$mbox = imap_open("{localhost:143}INBOX","user_id","password");
?>
```

Pour se connecter à un serveur **POP3** qui fonctionne sur le port 110 de la machine locale on peut utiliser la commande suivante :

```
<?php
$mbox = imap_open("{localhost/pop3:110}INBOX","user_id","password");
?>
```

Pour se connecter à un serveur **NNTP** qui fonctionne sur le port 119 de la machine locale on peut utiliser la commande:

```
<?php
$nntp = imap_open("{localhost/nntp:119}comp.test","", "");
?>
```

Pour se connecter à un serveur distant, remplacez "localhost" par le nom ou l'adresse IP de la machine.

[10.33.25 imap_ping](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`int imap_ping(int imap_stream)`
[PHP 3, PHP 4]

Retourne TRUE si le flot existe toujours, et FALSE sinon.

[imap_ping\(\)](#) vérifie que le flot **IMAP** est toujours actif, en lui envoyant un ping. Cette fonction permet de se rendre compte que du mail est arrivé : c'est même la méthode préconisée pour des tests périodiques de vérification du courrier. Cette fonction peut aussi servir à garder une connexion ouverte, avec les serveurs dotés d'un délai d'expiration.

Exemple avec `imap_open()`

```
<?php
$mbox = imap_open ("{your.imap.host:143}", "username", "password");
echo "<p><h1>Mailboxes</h1>\n";
$folders = imap_listmailbox ($mbox, "{your.imap.host:143}", "*");
if ($folders == FALSE) {
echo "Call failed<br>\n";
} else {
while (list ($key, $val) = each ($folders)) {
echo $val."<br>\n";
}
```

```

    }
}
echo "<p><h1>Headers in INBOX</h1>\n";
$headers = imap_headers ($mbox);
if ($headers == FALSE) {
    echo "Call failed<br>\n";
} else {
    while (list ($key,$val) = each ($headers)) {
        echo $val."<br>\n";
    }
}
imap_close($mbox);
?>

```

10.33.26 imap_renamemailbox

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_renamemailbox](#) (int *imap_stream*, string *old_mbox*, string *new_mbox*)
[PHP 3, PHP 4]

[imap_renamemailbox\(\)](#) renomme la boîte aux lettres *old_mbox* en *new_mbox*.

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

Voir aussi [imap_createmailbox\(\)](#), [imap_deletemailbox\(\)](#) et [imap_open\(\)](#) pour le format de *mbox*.

10.33.27 imap_reopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_reopen](#) (string *imap_stream*, string *mailbox*, string *flags*)
[PHP 3, PHP 4]

[imap_reopen\(\)](#) réouvre la connexion spécifiée au serveur **IMAP** ou **NNTP**, avec une nouvelle boîtes aux lettres.

Les options sont des masques de bit, qui peuvent contenir les valeurs suivantes :

- OP_READONLY – Ouvre une boîte aux lettres en lecture seule
- OP_ANONYMOUS – Ne pas utiliser, ou modifier le fichier .newsrsrc pour les news
- OP_HALFOPEN – Pour les noms **IMAP** et **NNTP**, ouvre une connexion mais n'ouvre pas une boîte aux lettres.
- CL_EXPUNGE – Supprime automatiquement la boîte aux lettres de la liste, lors de la terminaison du flot. (voir [imap_delete\(\)](#) et [imap_expunge\(\)](#)).

10.33.28 imap_subscribe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_subscribe](#) (int *imap_stream*, string *mbox*)
[PHP 3, PHP 4]

[imap_subscribe\(\)](#) souscrit à la boîte aux lettres *mbox*.

[imap_subscribe\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

[10.33.29 imap_undelete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_undelete](#) (int *imap_stream*, int *msg_number*)

[PHP 3, PHP 4]

[imap_undelete\(\)](#) enlève la marque d'effacement du message *msg_number*, placée avec [imap_delete\(\)](#).

[imap_subscribe\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[10.33.30 imap_unsubscribe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_unsubscribe](#) (int *imap_stream*, string *mbox*)

[PHP 3, PHP 4]

[imap_unsubscribe\(\)](#) termine la souscription à la boîte aux lettres *mbox*.

[imap_unsubscribe\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[10.33.31 imap_qprint](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_qprint](#) (string *string*)

[PHP 3, PHP 4]

[imap_qprint\(\)](#) convertit la chaîne à guillemets *string* en une chaîne à 8 bits.

Retourne une chaîne 8 bits (binaire).

Voir aussi [imap_8bit\(\)](#).

[10.33.32 imap_8bit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_8bit](#) (string *string*)

[PHP 3, PHP 4]

[imap_8bit\(\)](#) convertit la chaîne à 8 bits en une chaîne à guillemets.

Retourne une chaîne à guillemets.

Voir aussi [imap_qprint\(\)](#).

[10.33.33 imap_binary](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_binary](#) (string *string*)

[PHP 3>= 3.0.2, PHP 4]

[imap_binary\(\)](#) convertit la chaîne à 8 bits *string* en une chaîne à base64.
 Retourne la chaîne codée.
 Voir aussi [imap_base64\(\)](#).

10.33.34 [imap_scanmailbox](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_scanmailbox](#) (int *imap_stream*, string *string*)
 [PHP 3, PHP 4]

[imap_scanmailbox\(\)](#) retourne un tableau contenant les noms des boîtes aux lettres qui contiennent la chaîne *string*. [imap_scanmailbox\(\)](#) est simialaire à [imap_listmailbox\(\)](#), mais va aussi rechercher la chaîne *string* dans les données de la boîte aux lettres. Reportez vous à [imap_getmailboxes\(\)](#) pour une description des paramètres *ref* et *pattern*.

10.33.35 [imap_mailboxmsginfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_mailboxmsginfo](#) (int *imap_stream*)
 [PHP 3>= 3.0.2, PHP 4]

[imap_mailboxmsginfo\(\)](#) retourne les informations à propos de la boîte aux lettres courante. Retourne FALSE en cas d'échec.

[imap_mailboxmsginfo\(\)](#) vérifie le statut courant de la boîte aux lettres sur le serveur, et retourne un objet avec les propriétés suivantes :

Date	Date de dernière modification du contenu de la boîte aux lettres @tab
Driver	Pilote
Mailbox	Nom de la boîte aux lettres
Nmsgs	Nombre de messages
Recent	Nombre de messages récents
Unread	Nombre de messages non lus
Deleted	Nombre de messages effacés
Size	Taille de la boîte aux lettres

Exemple avec [imap_mailboxmsginfo\(\)](#)

```
<?php
$mbx = imap_open("{your.imap.host}INBOX","utilisateur", "mot_de_passe")
|| die("conexion impossible: ".imap_last_error());
$check = imap_mailboxmsginfo($mbx);
if($check) {
print "Date: "      . $check->Date      . "<br>\n" ;
print "Pilote: "    . $check->Driver    . "<br>\n" ;
print "Mailbox: "   . $check->Mailbox   . "<br>\n" ;
}
```

```

print "Messages: " . $check->Nmsgs . "<br>\n" ;
print "Récent: " . $check->Recent . "<br>\n" ;
print "Non lus: " . $check->Unread . "<br>\n" ;
print "Effacés: " . $check->Deleted . "<br>\n" ;
print "Taille: " . $check->Size . "<br>\n" ;
} else {
print "imap_check() a échoué: ".imap_last_error(). "<br>\n";
}
imap_close($mbox);
?>

```

[10.33.36 imap_rfc822_write_address](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_rfc822_write_address](#)(string *mailbox*, string *host*, string *personal*)
 [PHP 3>= 3.0.2, PHP 4]

mailbox retourne une adresse email proprement formatée, à partir du nom de la boîte aux lettres de l'hôte *host*, et des informations personnelles *personal*.

Exemple avec imap_rfc822_write_address()

```

<?php
print imap_rfc822_write_address("hartmut","cvs.php.net","Hartmut Holzgraefe")."\n";
?>

```

[10.33.37 imap_rfc822_parse_adrlist](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_rfc822_parse_adrlist](#)(string *address*, string *default_host*)
 [PHP 3>= 3.0.2, PHP 4]

address analyse la chaîne *address* et essaie, pour chaque adresse, de retourner un tableau d'objets. Les 4 objets sont :

- mailbox – Le nom de la boîte aux lettres
- host – le nom de l'hôte
- personal – Le nom personnel
- adl – at domain source route (NDT : ???).

Exemple avec imap_rfc822_parse_adrlist()

```

<?php
$address_string = "Hartmut Holzgraefe <hartmut@cvs.php.net>, postmaster@somedomain.net, root";
$address_array = imap_rfc822_parse_adrlist($address_string,"somedomain.net");
if(! is_array($address_array)) die("une erreur...\n");

```

```

reset($address_array);
while(list($key,$val)=each($address_array)){
print "boîte    : ".$val->mailbox."<br>\n";
print "hôte    : ".$val->host."<br>\n";
print "personnel: ".$val->personal."<br>\n";
print "adl     : ".$val->adl."<p>\n";
}
?>

```

10.33.38 imap_setflag_full

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_setflag_full](#) (int *stream*, string *sequence*, string *flag*, string *options*)
 [PHP 3>= 3.0.3, PHP 4]

[imap_setflag_full\(\)](#) affecte le flag spécifié aux messages de la séquence donnée.

Les flags que vous pouvez modifier sont "\\Seen", "\\Answered", "\\Flagged", "\\Deleted", "\\Draft" et "\\Recent" (comme défini dans la RFC2060).

Les options sont un masque de bits, et peuvent contenir les valeurs suivantes :

ST_UID la séquence contient des UIDs au lieu de numéro de séquence.

Exemple avec imap_setflag_full()

```

<?php
$mbx = imap_open("{your.imap.host:143}", "utilisateur", "mot_de_passe")
|| die("can't connect: ".imap_last_error());
$status = imap_setflag_full($mbx, "2,5", "\\Seen \\Flagged");
print gettype($status)."\n";
print $status."\n";
imap_close($mbx);
?>

```

10.33.39 imap_clearflag_full

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_clearflag_full](#) (int *stream*, string *sequence*, string *flag*, string *options*)
 [PHP 3>= 3.0.3, PHP 4]

stream efface le flag spécifié dans les messages de la séquence *sequence*.

Les options sont un masque de bit, qui accepte les valeurs suivantes :

ST_UID : la séquence contient des UIDs au lieu de numéro de séquence

10.33.40 imap_sort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_sort](#)(int *stream*, int *criteria*, int *reverse*, int *options*)
[PHP 3>= 3.0.3, PHP 4]

[imap_sort\(\)](#) retourne un tableau de nombre de message, triés suivant les paramètres suivants :

reverse vaut 1 pour signifier : tri inverse.

Les critères peuvent être un (et un seul) parmi les suivants :

SORTDATE	Date du message
SORTARRIVAL	Date d'arrivée
SORTFROM	Nom de la première boîte aux lettres de l'adresse d'origine (From address)
SORTSUBJECT	Sujet du message
SORTTO	Nom de la première boîte aux lettres de destination (To address)
SORTCC	Nom de la boîte aux lettres de copie cachée (cc address)
SORTSIZE	Taille du message en octets

Les flags dont des masques de bits, d'un ou plusieurs des éléments suivants :

SE_UID	Retourne l'UIDs à la place d'une séquence de nombres.
SE_NOPREFETCH	Ne pas pré-télécharger les messages trouvés.

10.33.41 imap_fetchheader

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_fetchheader](#)(int *imap_stream*, int *msgno*, int *flags*)
[PHP 3>= 3.0.3, PHP 4]

[imap_fetchheader\(\)](#) retourne l'entête brut et complet RFC 822 du message *msgno*, et le retourne sous la forme d'une chaîne.

Les options sont :

FT_UID	L'argument <i>msgno</i> est un UID
FT_INTERNAL	la chaîne renvoyée est au format "internal" , c'est à dire sans canonisation des CRLF
FT_PREFETCHTEXT	RFC822.TEXT doit être pré téléchargé en même temps que l'entête. Cela réduit le RTT sur une connexion IMAP , si le message complet est souhaité. (e.g. dans une opération de sauvegarde dans un fichier).

10.33.42 imap_uid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_uid](#)(int *imap_stream*, int *msgno*)

[PHP 3>= 3.0.3, PHP 4]

[imap_uid\(\)](#) retourne l'UID pour le message *msgno*. Un UID est un identifiant unique que ne change jamais, alors que le numéro du message dans la liste des messages peut changer à toute modification de la boîte aux lettres. C'est la fonction contraire de [imap_msgno\(\)](#).

10.33.43 [imap_msgno](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [imap_msgno](#)(int *imap_stream*, int *uid*)

[PHP 3>= 3.0.3, PHP 4]

[imap_msgno\(\)](#) retourne le numéro de séquence de message pour l'UID *uid*. C'est la fonction contraire de [imap_uid\(\)](#).

10.33.44 [imap_search](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_search](#)(int *imap_stream*, string *criteria*, int *flags*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b4]

[imap_search\(\)](#) effectue une recherche dans la boîte aux lettres courantes, sur le flot **IMAP** courant.

criteria est une chaîne, délimitée par des espaces, dans laquelle les mots-clés suivants sont acceptés. Tous les arguments multi-mots doivent être entre guillemets :

- ALL – retourne tous les message qui vérifie le reste du critère.
- ANSWERED – tous les messages avec le flag `\\ANSWERED`
- BCC "string" – tous les messages avec la chaîne "string" dans le champs Bcc:
- BEFORE "date" – tous les messages avec Date: avant "date"
- BODY "string" – tous les messages avec "string" dans le corps
- CC "string" – tous les messages avec "string" dans le champs Cc:
- DELETED – tous les messages effacés
- FLAGGED – tous les messages avec le flag `\\FLAGGED` (parfois interprété comme Important ou Urgent)
- FROM "string" – tous les messages avec la chaîne "string" dans le champs From:
- KEYWORD "string" – tous les messages avec la chaîne "string" comme mot clé
- NEW – tous les nouveaux messages
- OLD – tous les anciens messages
- ON "date" – tous les messages avec la date "date" comme champs Date:
- RECENT – tous les messages avec le flag `\\RECENT`
- SEEN – tous les messages lus (avec le flag `\\SEEN` flag)
- SINCE "date" – tous les messages avec la date Date: après "date"
- SUBJECT "string" – tous les messages avec la chaîne "string" dans le champs Subject:
- TEXT "string" – tous les messages avec le texte "string"
- TO "string" – tous les messages avec la chaîne "string" dans le champs To:
- UNANSWERED – tous les messages non répondus
- UNDELETED – tous les messages non effacés
- UNFLAGGED – tous les messages non flaggés

- UNKEYWORD "string" – tous les messages dans le mot clés "string"
- UNSEEN – tous les messages non lus

Par exemple, pour rechercher les messages non répondus, envoyés par maman, vous pouvez utiliser : "UNANSWERED FROM maman". Les recherches semblent insensibles à la casse. Cette liste de critères est issue du code d'un client C UW et peut être incomplète ou imprécise. (voir aussi RFC2060, section 6.4.4). Les valeurs pour les flags sont SE_UID, qui fait que le tableau réponse contient les UIDs plutôt que les numéros de séquence.

10.33.45 imap_last_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_last_error](#) (void)
[PHP 3>= 3.0.12, PHP 4 >= 4.0b4]

[imap_last_error\(\)](#) retourne le texte complet de la dernière erreur **IMAP** (si elle existe) qui est survenu lors de la dernière requête. La pile d'erreur n'est pas touchée. Appeler [imap_last_error\(\)](#) successivement dans nouvelles erreurs retournera la même erreur.

10.33.46 imap_errors

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_errors](#) (void)
[PHP 3>= 3.0.12, PHP 4 >= 4.0b4]

[imap_errors\(\)](#) retourne tous les messages d'erreurs **IMAP** générés depuis le dernier appel à [imap_errors\(\)](#), ou depuis le début de la page. Lorsque [imap_errors\(\)](#) est appelés, la pile d'erreur est vidée.

10.33.47 imap_alerts

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_alerts](#) (void)
[PHP 3>= 3.0.12, PHP 4 >= 4.0b4]

[imap_alerts\(\)](#) retourne tous les messages d'alerte **IMAP** générés depuis le dernier appel à [imap_alerts\(\)](#) ou depuis le début de la page. Lorsque [imap_alerts\(\)](#) est appelé, la pile d'alertes est vidée.

10.33.48 imap_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [imap_status](#) (int *imap_stream*, string *mailbox*, int *options*)
[PHP 3>= 3.0.4, PHP 4]

[imap_status\(\)](#) retourne un objet contenant les informations de statut. Les options valables sont :

- SA_MESSAGES – met la valeur de status->messages au nombre de messages dans la boîtes aux lettres.

- SA_RECENT – met la valeur de status->recent au nombre de messages récents dans la boîte aux lettres.
- SA_UNSEEN – met la valeur de status->unseen au nombre de messages non lus dans la boîte aux lettres.
- SA_UIDNEXT – met la valeur de status->uidnext à la prochaine valeur d'uid qui sera utilisée.
- SA_UIDVALIDITY – met la valeur de status->uidvalidity à une constante, qui change lorsque l'uid de la boîte aux lettres n'est plus valide.
- SA_ALL – fixe les valeurs de toutes les précédentes.

status->flags est aussi fixé : c'est un masque de bit qui peut contenir tous les flags ci dessus.

Exemple imap_status()

```
<?php
$mailbox = imap_open("{your.imap.host}", "utilisateur", "mot_de_passe", OP_HALFOPEN)
|| die("can't connect: ".imap_last_error());
$status = imap_status($mailbox, "{your.imap.host}INBOX", SA_ALL);
if($status) {
    print("Messages:      ". $status->messages      )."<br>\n";
    print("Récents:      ". $status->recent          )."<br>\n";
    print("Non lus:      ". $status->unseen           )."<br>\n";
    print("UIDnext:      ". $status->uidnext          )."<br>\n";
    print("UIDvalidité: ". $status->uidvalidity)."<br>\n";
} else
    print "imap_status a échoué : ".imap_lasterror()."\n";
imap_close($mailbox);
?>
```

10.33.49 imap_utf7_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_utf7_decode](#)(string *text*)

[PHP 3>= 3.0.15, PHP 4 >= 4.0b4]

[imap_utf7_decode\(\)](#) décode la chaîne UTF-7 *text* en données 8 bits.

[imap_utf7_decode\(\)](#) retourne les données 8bits décodées, ou FALSE si la chaîne *text* n'est pas au format UTF-7. Cette fonction sert à décoder des noms de boîtes aux lettres qui contiennent des caractères internationaux hors de l'espace ASCII. Le standard UTF-7 est défini dans la [RFC 2060](#), section 5.1.3 (l'original UTF-7 a été défini dans [RFC1642](#)).

10.33.50 imap_utf7_encode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_utf7_encode](#)(string *data*)

[PHP 3>= 3.0.15, PHP 4 >= 4.0b4]

[imap_utf7_encode\(\)](#) retourne les données *data* 8bits encodées, ou FALSE si une erreur est survenue. Cette fonction sert à encoder des noms de boîtes aux lettres qui contiennent des caractères internationaux hors de l'espace ASCII. Le standard UTF-7 est défini dans la [RFC 2060](#), section 5.1.3 (l'original UTF-7 a été défini

dans [RFC1642](#)).

Retourne un texte UTF-7.

[10.33.51 imap_utf8](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_utf8](#)(string *text*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0RC1]

[imap_utf8\(\)](#) convertit le texte *text* en UTF8 (comme défini dans [RFC2044](#)).

[10.33.52 imap_fetch_overview](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_fetch_overview](#)(int *imap_stream*, string *sequence*, int *flags*)

[PHP 3>= 3.0.4, PHP 4]

[imap_fetch_overview\(\)](#) lit les entêtes des messages mails de la séquence *sequence* et retourne un sommaire de leur contenu. *sequence* va contenir une séquence d'indice de message ou d'UIDs, si *flags* cotient FT_UID. La valeur retournée est un tableau d'objets, un par message d'entête décrit :

- subject – Le sujet du message
- from – Expéditeur
- date – Date d'expédition
- message_id – Identification du message
- references – est une référence sur l'id de ce message
- size – taille en octets
- uid – UID du message dans la boîte aux lettres
- msgno – numéro de séquence du message dans la boîte
- recent – Ce message est récent
- flagged – Ce message est marqué
- answered – Ce message a donné lieu à une réponse
- deleted – Ce message est marqué pour l'effacement
- seen – Ce message est déjà lu
- draft – Ce message est un brouillon

Exemple avec [imap_fetch_overview\(\)](#)

```
<?php
$mbx = imap_open("{your.imap.host:143}", "utilisateur", "mot_de_passe")
|| die("connexion impossible : ".imap_last_error());
$overview = imap_fetch_overview($mbx, "2,4:6", 0);
if(is_array($overview)) {
    reset($overview);
    while( list($key,$val) = each($overview)) {
        print      $val->msgno
                . " - " . $val->date
                . " - " . $val->subject
                . "\n";
    }
}
```

```

    }
}
imap_close($mbox);
?>

```

10.33.53 imap_mime_header_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [imap_header_decode](#) (string *text*)

@xref{function.imap-mime-header-decode, , imap_mime_header_decode()} décode un message MIME qui contient des données non ASCII (Voir [RFC2047](#)) Les éléments décodés sont retournés dans un tableau d'objets. Chacun de ces objets a deux propriétés : "charset" & "text". Si l'élément n'a pas été encodé, ou, en d'autres termes, s'il est en clair (plain US_ASCII), la propriété "charset" est mise à "default".

Exemple imap_mime_header_decode()

```

<?php
$text="=?ISO-8859-1?Q?Keld_J=F8rn_Simonsen?= <keld@dkuug.dk>";
$elements=imap_mime_header_decode($text);
for($i=0;$i<count($elements);$i++) {
echo "Charset: {$elements[$i]->charset}\n";
echo "Texte: {$elements[$i]->text}\n\n";
}
?>

```

Dans l'exemple ci dessus, on trouve deux éléments : le premier a été encodé en ISO-8859-1, et le second est en clair.

10.33.54 imap_mail_compose

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_mail_compose](#) (array *envelope*, array *body*)

[PHP 3>= 3.0.5, PHP 4]

Exemple imap_mail_compose()

```

<?php
$envelope["from"]="musone@afterfive.com";
$envelope["to"]="musone@darkstar";
$envelope["cc"]="musone@edgeglobal.com";
$part1["type"]=TYPEMULTIPART;
$part1["subtype"]="mixed";
$filename="/tmp/imap.c.gz";
$fp=fopen($filename,"r");
$contents=fread($fp,filesize($filename));
fclose($fp);
$part2["type"]=TYPEAPPLICATION;

```

```

$part2["encoding"] = ENCBINARY;
$part2["subtype"] = "octet-stream";
$part2["description"] = basename($filename);
$part2["contents.data"] = $contents;
$part3["type"] = TYPETEXT;
$part3["subtype"] = "plain";
$part3["description"] = "description3";
$part3["contents.data"] = "contents.data3\n\n\n\t";
$body[1] = $part1;
$body[2] = $part2;
$body[3] = $part3;
echo nl2br(imap_mail_compose($envelope,$body));
?>

```

[10.33.55 imap_mail](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [imap_mail](#)(string *to*, string *subject*, string *message*, string *additional_headers* , string *cc* , string *bcc* , string *rpath*)

[PHP 3 >= 3.0.14, PHP 4 >= 4.0b4]

[imap_mail\(\)](#) est uniquement disponible sous PHP 3.

[10.34 Options PHP & informations](#)

[\[Notes en ligne\]](#)

[10.34.1 assert](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [assert](#)(string)bool *assertion*|

[PHP 4 >= 4.0b4]

[assert\(\)](#) va vérifier l'assertion *assertion* et prendre la mesure appropriée si le résultat est FALSE.

Si *assertion* est donnée sous la forme d'une chaîne, elle sera évaluée comme un code PHP par la fonction [assert\(\)](#). Les avantages de ce type d'assertion sont d'être moins lourd si la vérification d'assertion est désactivée, et le contenu des messages lorsque l'assertion échoue.

Il est recommandé de n'utiliser les assertions que comme outil de débogage. Vous pouvez les utiliser pour les vérifications d'usage : ces conditions doivent normalement être vraie, et indiquer une erreur de programmation si ce n'est pas le cas. Vous pouvez aussi vérifier la présence de certaines extensions ou limitations du système.

Les assertions ne doivent pas être utilisées pour faire des opérations de vérifications en production, comme par exemple des vérifications de valeur d'arguments. En conditions normales, votre code doit être en état de fonctionner si la vérification d'assertion est désactivée.

Le comportement de [assert\(\)](#) peut être configuré par [assert_options\(\)](#) ou par .ini-settings.

10.34.2 assert-options

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [assert_options](#)(int *what*, mixed *value*)

[PHP 4 >= 4.0b4]

Avec [assert_options\(\)](#) vous pouvez modifier les diverses options de la fonction [assert\(\)](#), ou simplement connaître la configuration actuelle.

option	paramètre d'ini	valeur par défaut	description
ASSERT_ACTIVE	assert.active	1	active l'évaluation de la fonction assert()
ASSERT_WARNING	assert.warning	1	génère une alerte PHP pour chaque assertion fausse
ASSERT_BAIL	assert.bail	0	terminer l'exécution en cas d'assertion fausse
ASSERT_QUIET_EVAL	assert.quiet_eval	0	inactive le rapprot d'erreur durant l'évaluation d'une assertion @tab
ASSERT_CALLBACK	assert_callback	(null)	fonction utilisateur de traitement des assertions fausses

[assert_options\(\)](#) retourne la valeur originale de l'option, ou bien FALSE en cas d'erreur.

10.34.3 extension_loaded

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [extension_loaded](#)(string *name*)

[PHP 3 >= 3.0.10, PHP 4 >= 4.0b4]

[extension_loaded\(\)](#) retourne vraie si l'extension *name* a été chargée. Vous pouvez voir les différents noms des extensions, en utilisant la fonction [phpinfo\(\)](#).

Voir aussi [phpinfo\(\)](#). Note : Cette fonction a été ajoutée dans 3.0.10.

10.34.4 dl

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [dl](#)(string *library*)

[PHP 3, PHP 4]

Charge l'extension PHP *library* à la volée . Voir aussi la directive de configuration [7.1.5.2 ini.extension-dir](#).

[10.34.5 getenv](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getenv](#) (string *varname*)
[PHP 3, PHP 4]

Retourne la valeur de la variable d'environnement *varname*, ou FALSE en cas d'erreur.

```
<?php
$ip = getenv("REMOTE_ADDR"); // retourne l'adresse IP de l'utilisateur
?>
```

Vous pouvez voir une liste complète des variables d'environnement en utilisant la fonction [phpinfo\(\)](#). Vous pouvez trouver la signification de chacune d'entre elles en consultant le site concernant [CGI specification](#) (en anglais>, et particulièrement la page concernant les [variables d'environnement](#).

[10.34.6 get_cfg_var](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_cfg_var](#) (string *varname*)
[PHP 3, PHP 4]

Retourne la valeur courante de l'option PHP *varname*, ou bien FALSE en cas d'erreur.

[get_cfg_var\(\)](#) ne retourne pas les options qui ont été choisies lors de la compilation de PHP, ni ne lit dans le fichier de configuration d'Apache.

Pour vérifier si le système utilise le [7.1 Le fichier de configuration](#), essayez de lire la valeur de `cfg_file_path`. Si cette valeur est disponible, alors le fichier de configuration est utilisé.

[10.34.7 get_current_user](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_current_user](#) (void)
[PHP 3, PHP 4]

Retourne le nom du possesseur du script courant.

Voir aussi [getmyuid\(\)](#), [getmypid\(\)](#), [getmyinode\(\)](#), et [getlastmod\(\)](#).

[10.34.8 get_magic_quotes_gpc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

long [get_magic_quotes_gpc](#) (void)
[PHP 3>= 3.0.6, PHP 4]

Retourne la configuration actuelle de l'option [7.1.1.17 ini.magic-quotes-gpc](#) (0 pour l'option désactivée, 1 pour l'option activée).

Voir aussi [get_magic_quotes_runtime\(\)](#), [set_magic_quotes_runtime\(\)](#).

10.34.9 get_magic_quotes_runtime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

long [get_magic_quotes_runtime](#)(void)

[PHP 3>= 3.0.6, PHP 4]

RRetourne la configuration actuelle de l'option [7.1.1.18 ini.magic-quotes-runtime](#). (0 pour option désactivée, 1 pour option activée).

Voir aussi [get_magic_quotes_gpc\(\)](#), [set_magic_quotes_runtime\(\)](#).

10.34.10 getlastmod

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getlastmod](#)(void)

[PHP 3, PHP 4]

[getlastmod\(\)](#) retourne la date de dernière modification de la page. La valeur retournée est un marqueur de temps UNIX, utilisable comme paramètre avec la fonction [date\(\)](#). Retourne FALSE en cas d'erreur.

Exemple avec [getlastmod\(\)](#)

```
<?php
// affiche 'Dernière modification: March 04 1998 20:43:59.'
echo "Dernière modification: ".date( "F d Y H:i:s.", getlastmod() );
?>
```

Voir aussi [date\(\)](#), [getmyuid\(\)](#), [get_current_user\(\)](#), [getmyinode\(\)](#) et [getmypid\(\)](#).

10.34.11 getmyinode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmyinode](#)(void)

[PHP 3, PHP 4]

Retourne l'inode du script, ou FALSE en cas d'erreur.

Voir aussi [getmyuid\(\)](#), [get_current_user\(\)](#), [getmypid\(\)](#), et [getlastmod\(\)](#).

Note : Cette fonction est inopérante sur les systèmes Windows.

10.34.12 getmypid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmypid](#)(void)

[PHP 3, PHP 4]

Retourne le numéro de processus actuel ou FALSE en cas d'erreur.

Il est à noter que si vous utilisez PHP comme module Apache, il n'est pas garanti que deux invocations

distinctes de la fonction donnent des résultats différents.

Les identifiants de processus ne sont pas uniques, et forment une source d'entropie faible. Nous recommandons de ne pas utiliser les pid pour assurer la sécurité d'un système.

Voir aussi [getmyuid\(\)](#), [get_current_user\(\)](#), [getmyinode\(\)](#), et [getlastmod\(\)](#).

10.34.13 getmyuid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmyuid](#)(void)

[PHP 3, PHP 4]

Retourne l'UID du propriétaire du script actuel ou FALSE en cas d'erreur.

Voir aussi [getmypid\(\)](#), [get_current_user\(\)](#), [getmyinode\(\)](#), et [getlastmod\(\)](#).

10.34.14 getrusage

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [getrusage](#)(int *who*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[getrusage\(\)](#) est une interface à la fonction system `getrusage(2)`. Elle retourne un tableau associatif contenant les informations renvoyées par cet appel système. Si "who is 1", `getrusage` sera appelé avec le paramètre `RUSAGE_CHILDREN`.

Toutes les valeurs du tableau sont accessibles en utilisant leur nom dans le tableau.

Exemple `getrusage`

```
<?php
$dat = getrusage();
echo $dat["ru_nswap"];           # Taille de la mémoire swap
echo $dat["ru_majflt"];          # Nombre de page mémoires utilisées
echo $dat["ru_utime.tv_sec"];    # temps utilisateur (en secondes)
echo $dat["ru_utime.tv_usec"];  # temps utilisateur (en microsecondes)
?>
```

Consultez le manuel "man" pour plus de détails.

10.34.15 ini_alter

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ini_alter](#)(string *varname*, string *newvalue*)

[PHP 4]

Change la valeur de l'option de configuration *varname* et lui donne la valeur de *newvalue*. Retourne FALSE en cas d'échec, et la valeur précédente en cas de succès.

Note : Cette fonction est un alias de [ini_set\(\)](#)

Voir aussi [ini_get\(\)](#), [ini_restore\(\)](#), [ini_set\(\)](#)

10.34.16 ini_get

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ini_get](#) (string *varname*)
[PHP 4]

Retourne la valeur de l'option de configuration *varname* en cas de succès, et FALSE.
Voir aussi [ini_alter\(\)](#), [ini_restore\(\)](#), [ini_set\(\)](#)

10.34.17 ini_restore

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ini_restore](#) (string *varname*)
[PHP 4]

Restaure la valeur originale de l'option de configuration *varname*.
Voir aussi [ini_alter\(\)](#), [ini_get\(\)](#), [ini_set\(\)](#)

10.34.18 ini_set

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ini_set](#) (string *varname*, string *newvalue*)
[PHP 4 >= 4.0RC1]

Change la valeur de l'option de configuration *varname* et lui donne la valeur de *newvalue*. Retourne FALSE en cas d'échec, et la valeur précédente en cas de succès.
Voir aussi [ini_alter\(\)](#), [ini_get\(\)](#), [ini_restore\(\)](#)

10.34.19 phpcredits

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [phpcredits](#) (int *flag*)
[PHP 4]

[phpcredits\(\)](#) affiche la liste des développeurs PHP, des modules, etc... Elle génère le code HTML approprié pour insérer les informations dans une page. Le paramètre *flag* indique les informations qui doivent être affichées. Par exemple, pour afficher les crédits généraux, vous pouvez utiliser le code suivant :

```
<?php
phpcredits(CREDITS_GENERAL) ;
?>
```

Et pour afficher la liste des développeurs et du groupe de documentation dans une page séparée, vous utiliserez

```
<?php
phpcredits(CREDITS_GROUP + CREDITS_DOCS + CREDITS_FULLPAGE) ;
?>
```

Si vous vous sentez l'envie de placer tous les crédits dans votre page, vous pouvez utiliser ceci :

```
<html>
  <head>
    <title>Ma page de crédits</title>
  </head>
  <body>
    <?php
      // Un peu de votre code
phpcredits(CREDITS_ALL + CREDITS_FULLPAGE);
      // Un autre peu de votre code
    ?>
  </body>
</html>
```

Nom	Description
CREDITS_ALL	Tous les crédits, équivalent à : CREDITS_DOCS + CREDITS_GENERAL + CREDITS_GROUP + CREDITS_MODULES + CREDITS_FULLPAGE. La fonction génère alors une page HTML complète. @tab
CREDITS_DOCS	Les crédits du groupe de documentation
CREDITS_FULLPAGE	En général, ce paramètre est utilisé avec d'autres constantes. Il indique que la page ainsi générée doit être une page HTML complète, avec toutes les balises nécessaires. @tab
CREDITS_GENERAL	Crédits Généraux : conception et design du langage, auteurs de PHP 4.0, module SAPI. @tab
CREDITS_GROUP	Une liste des développeurs principaux
CREDITS_MODULES	Une liste des extensions de PHP, et leurs auteurs
CREDITS_SAPI	Cette constante est définie, mais elle n'est toujours pas utilisée sous PHP 4.0.1p12. @tab

Voir aussi [phpinfo\(\)](#), [phpversion\(\)](#), [php_logo_guid\(\)](#).

[10.34.20 phpinfo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [phpinfo](#) (int *what*)

[PHP 3, PHP 4]

Affiche de nombreuses informations sur le PHP, concernant sa configuration courante : options de compilation, extensions, version, informations sur le serveur, et environnement (lorsque compilé comme module), environnement PHP, chemins, utilisateur, entêtes HTTP, et licence GNU Public License. Les affichages peuvent être personnalisés en passant une ou plusieurs valeurs parmi les suivantes, comme paramètre optionnel *what* :

- INFO_GENERAL
- INFO_CREDITS
- INFO_CONFIGURATION
- INFO_MODULES
- INFO_ENVIRONMENT
- INFO_VARIABLES
- INFO_LICENSE
- INFO_ALL

Voir aussi [phpversion\(\)](#), [phpcredits\(\)](#) et [php_logo_guid\(\)](#).

[10.34.21 phpversion](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [phpversion](#) (void)

[PHP 3, PHP 4]

[phpversion\(\)](#) retourne le numéro de la version courante de PHP.

exemple de la fonction [phpversion\(\)](#)

```
<?php
// affiche le numéro de version courante du PHP.
echo "PHP Version: ".phpversion();
?>
```

Voir aussi [phpinfo\(\)](#).

[10.34.22 php_logo_guid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [php_logo_guid](#) (void)

[PHP 4 >= 4.0b4]

Note : Cette fonctionnalité a été ajoutée dans PHP 4 Beta 4.

10.34.23 php_sapi_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [php_sapi_name](#)

[PHP 4 >= 4.0.1]

[php_sapi_name\(\)](#) retourne une chaîne en minuscule qui décrit le type d'interface utilisé en le serveur web et PHP (Server API, SAPI). En CGI PHP, cette chaîne est "cgi", en mod_php pour Apache, cette chaîne est "apache", etc...

Exemple php_sapi_name()

```
<?php
$inter_type = php_sapi_name();
if ($inter_type == "cgi")
print "Vous utilisez CGI PHP\n";
else
print "Vous n'utilisez pas CGI PHP\n";
?>
```

10.34.24 php_uname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [php_uname](#)

[PHP 4 >= 4.0.2]

[php_uname\(\)](#) retourne les informations sur le système d'exploitation sur lequel tourne PHP.

Exemple php_uname()

```
<?php
if (substr/php_uname(), 0, 7) == "Windows") {
die("Désolé, ce script ne fonctionne pas sous Windows.\n");
}
?>
```

10.34.25 putenv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [putenv](#)(string *setting*)

[PHP 3, PHP 4]

Fixe la valeur d'une variable d'environnement.

Fixer la valeur d'une variable d'environnement

```
<?php
```

```
putenv("UNIQID=$uniqid");
?>
```

10.34.26 set_magic_quotes_runtime

[\[Notes en ligne\]](#) [\[Exemples\]](#)

long [set_magic_quotes_runtime](#)(int *new_setting*)
[PHP 3 >= 3.0.6, PHP 4]

Active/désactive l'option [7.1.1.18 ini.magic-quotes-runtime](#). (0 l'option est désactivée, 1 l'option est activée). Voir aussi [get_magic_quotes_gpc\(\)](#) et [get_magic_quotes_runtime\(\)](#).

10.34.27 set_time_limit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [set_time_limit](#)(int *seconds*)
[PHP 3, PHP 4]

[set_time_limit\(\)](#) fixe le délai d'expiration d'un script, en secondes. Si cette limite est atteinte, le script s'interrompt, et renvoie une erreur fatale. La valeur par défaut est 30 secondes ou, si c'est le cas, la valeur de la directive `max_execution_time` définie dans le [7.1 Le fichier de configuration](#). Si la valeur est zéro, il n'y a alors aucune limite imposée.

Lorsqu'elle est appelée, la fonction [set_time_limit\(\)](#) remet le compteur de zéro. En d'autres termes, si la limite par défaut est à 30 secondes, et qu'après 25 secondes d'exécution du script l'appel `set_time_limit(20)` est fait, alors le script tournera pendant un total de 45 secondes avant de finir.

Notez que [set_time_limit\(\)](#) n'a pas d'effet lorsque PHP fonctionne en mode [7.1.3.1 ini.safe-mode](#). Il n'y a pas d'autre solution que de changer de mode, ou de modifier la durée maximale d'exécution dans le [7.1 Le fichier de configuration](#).

10.34.28 zend_logo_guid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [zend_logo_guid](#)(void)
[PHP 4 >= 4.0b4]

Note : Cette fonctionnalité a été ajoutée en PHP 4 Beta 4.

10.34.29 get_loaded_extensions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_loaded_extensions](#)(void)
[PHP 4 >= 4.0b4]

[get_loaded_extensions\(\)](#) retourne un tableau contenant les noms de tous les modules compilés et chargés sur l'interpréteur PHP courant.

Par exemple, la ligne ci dessous

```
<?php
print_r(get_loaded_extensions());
?>
```

affichera la liste suivante : Array ([0] => xml [1] => wddx [2] => standard [3] => session [4] => posix [5] => pgsql [6] => pcre [7] => gd [8] => ftp [9] => db [10] => Calendar [11] => bcmath)

Voir aussi: [get_extension_funcs\(\)](#).

10.34.30 get_extension_funcs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_extension_funcs](#)(string *module_name*)

[PHP 4 >= 4.0b4]

[get_extension_funcs\(\)](#) retourne le nom des des fonctions définies dans le module *module_name*.

Par exemple, les lignes suivantes :

```
<?php
print_r(get_extension_funcs("xml"));
print_r(get_extension_funcs("gd"));
?>
```

vont afficher la liste des fonctions disponibles avec les modules *xml* et *gd*.

Voir aussi: [get_loaded_extensions\(\)](#)

10.34.31 get_required_files

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_required_files](#)(void)

[PHP 4 >= 4.0RC2]

[get_required_files\(\)](#) retourne un tableau associatif contenant les noms de tous les fichiers qui ont été chargés dans un script avec la fonction [require_once\(\)](#). Les index de ces tableaux sont les noms des fichiers utilisés dans la fonction [require_once\(\)](#) sans les extensions ".php".

L'exemple ci dessous

Afficher les fichiers requis et inclus

```
<?php
require_once("local.php");
require_once("../inc/global.php");
for ($i=1; $i<5; $i++)
include "util".$i.".php";
echo "Fichiers requis par require_once()\n";
print_r(get_required_files());
echo "Fichiers inclus par included_once()\n";
print_r(get_included_files());
?>
```

va générer l'affichage suivant : Fichiers requis par require_once() Array ([local]

```
=> local.php [../inc/global] => /full/path/to/inc/global.php ) Fichiers
inclus par included_once() Array ( [util1] => util1.php [util2] =>
util2.php [util3] => util3.php [util4] => util4.php )
```

Note : Etant donné qu'en PHP 4.0.1pl2, cette fonction suppose que les noms de fichiers **required_once** se terminent par l'extension ".php", les autres extensions ne fonctionneront pas.

Voir aussi: [require_once\(\)](#), [include_once\(\)](#), [get_included_files\(\)](#)

[10.34.32 get_included_files](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_included_files](#)(void)
[PHP 4 >= 4.0RC1]

[get_required_files\(\)](#) retourne un tableau associatif contenant les noms de tous les fichiers qui ont été chargés dans un script avec la fonction [include_once\(\)](#). Les index de ces tableaux sont les noms des fichiers utilisés dans la fonction [include_once\(\)](#) sans les extensions ".php".

Note : Etant donné qu'en PHP 4.0.1pl2, cette fonction suppose que les noms de fichiers **required_once** se terminent par l'extension ".php", les autres extensions ne fonctionneront pas.

Voir aussi: [require_once\(\)](#), [include_once\(\)](#), [get_required_files\(\)](#)

[10.35 Ingres II](#)

[\[Notes en ligne\]](#)

Ces fonctions permettent l'accès à un serveur de base de données Ingres II.

Pour pouvoir utiliser ces fonctions, vous devez compiler PHP avec le support Ingres, en utilisant l'option `--with-ingres`. Ceci nécessite les fichiers de bibliothèque de l'en-tête d'Open API qui sont inclus dans Ingres II. Si la variable d'environnement `II_SYSTEM` n'est pas correctement initialisée vous devrez utiliser `--with-ingres=REP` pour spécifier le répertoire où a été installé Ingres.

Lorsque cette extension est utilisée avec Apache, si Apache ne démarre pas et émet l'erreur "PHP Fatal error: Unable to start ingres_ii module in Unknown on line 0", assurez-vous que la variable d'environnement `II_SYSTEM` est correctement initialisée. Il suffit souvent d'ajouter `"export II_SYSTEM="/home/ingres/II"` dans le script qui démarre Apache, juste avant le lancement de httpd.

Note : Si vous avez déjà utilisé des extensions PHP permettant l'accès à d'autres serveurs de bases de données, notez qu'Ingres n'accepte pas de requêtes et/ou de transactions concurrentes sur la même connexion, et donc vous ne trouverez aucun identifiant de résultat ou de transaction dans cette extension. Le résultat d'une requête doit être traité avant d'envoyer une autre requête, et une transaction doit être validée ("commit") ou annulée ("roll back") avant de pouvoir en ouvrir une nouvelle (l'ouverture de transaction est fait automatiquement à l'envoi de la première requête).

[10.35.1 ingres_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [ingres_connect](#)(string *database* , string *username* , string *password*)
[PHP 4 >= 4.0.2]

[ingres_connect\(\)](#) retourne une ressource représentant un lien Ingres II en cas de succès, et FALSE sinon.

[ingres_connect\(\)](#) établit une connexion avec la base de données Ingres désignée par *database*, qui suit la syntaxe *[node_id::]dbname[/svr_class]*.

Si certains paramètres sont manquants, [ingres_connect\(\)](#) utilise les valeurs de *ingres.default_database*, *ingres.default_user* et *ingres.default_password* indiquées dans ``php.ini'`.

La connexion est fermée lorsque le script se termine ou en cas d'appel à [ingres_close\(\)](#).

Toutes les autres fonctions Ingres utilisent le dernier lien ouvert comme lien par défaut, il n'est donc nécessaire de conserver la valeur de retour qu'en cas d'utilisation de plus d'un lien en même temps.

Exemple pour ingres_connect()

```
<?php
    $link = ingres_connect("mydb", "user", "pass")
or die("Erreur de connexion");
print("Connexion réussie");
ingres_close($link);
?>
```

Exemple pour ingres_connect() utilisant le lien par défaut

```
<?php
ingres_connect("madb", "user", "pass")
or die("Erreur de connexion");
print("Connexion réussie");
ingres_close();
?>
```

Voir aussi [ingres_pconnect\(\)](#), et [ingres_close\(\)](#).

10.35.2 ingres_pconnect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [ingres_pconnect](#) (string *database* , string *username* , string *password*)
[PHP 4 >= 4.0.2]

[ingres_pconnect\(\)](#) retourne une ressource représentant un lien (link) Ingres II en cas de succès, et FALSE sinon.

Voir [ingres_connect\(\)](#) pour le détail des paramètres et des exemples. Il n'y a que 2 différences entre [ingres_pconnect\(\)](#) et [ingres_connect\(\)](#): Tout d'abord, à la connexion, la fonction cherche un lien persistant déjà ouvert avec les mêmes paramètres. Si un tel lien est trouvé, un identificateur pour ce lien est retourné au lieu d'établir une nouvelle connexion. Ensuite, la connexion vers le serveur Ingres n'est pas fermée lorsque le script se termine. En fait, le lien reste ouvert pour pouvoir être réutilisé ([ingres_close\(\)](#) ne ferme pas les liens établis avec [ingres_pconnect\(\)](#)). C'est pourquoi ce type de lien est dit 'persistant'.

Voir aussi [ingres_connect\(\)](#) et [ingres_close\(\)](#).

10.35.3 ingres_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ingres_close](#) (resource *link*)
[PHP 4 >= 4.0.2]

[ingres_close\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ingres_close\(\)](#) ferme la connexion au serveur Ingres associée au lien spécifié. Si le paramètre *link* n'est pas spécifié, le dernier lien ouvert est utilisé.

Habituellement l'appel à [ingres_close\(\)](#) n'est pas nécessaire, car il ne ferme pas les connexions persistantes, et toutes les connexions non-persistantes sont automatiquement fermées à la fin du script.

Voir aussi [ingres_connect\(\)](#) et [ingres_pconnect\(\)](#).

10.35.4 ingres_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ingres_query](#) (string *query* , resource *link*)

[PHP 4 >= 4.0.2]

[ingres_query\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ingres_query\(\)](#) envoie la requête *query* au serveur Ingres. La requête doit être valide (voir le guide de référence SQL pour Ingres).

La requête s'ajoute à la transaction en cours. S'il n'y a pas de transaction ouverte, [ingres_query\(\)](#) en ouvre une nouvelle. Pour fermer une transaction, vous pouvez soit appeler [ingres_commit\(\)](#) pour valider les changements effectués sur la base de données ou [ingres_rollback\(\)](#) pour les annuler. Lorsque le script se termine, toute transaction ouverte est annulée (par appel à [ingres_rollback\(\)](#)). Vous pouvez aussi utiliser [ingres_autocommit\(\)](#) avant d'ouvrir une transaction pour que chaque requête SQL soit validée immédiatement et automatiquement.

Certains types de requêtes SQL ne peuvent pas être envoyées par cette fonction :

- CLOSE (voir [ingres_close\(\)](#)).
- COMMIT (voir [ingres_commit\(\)](#)).
- CONNECT (voir [ingres_connect\(\)](#)).
- DISCONNECT (voir [ingres_close\(\)](#)).
- get dbevent
- PREPARE TO COMMIT
- ROLLBACK (voir [ingres_rollback\(\)](#)).
- savepoint
- SET AUTOCOMMIT (voir [ingres_autocommit\(\)](#)).
- Les requêtes relatives aux curseurs ne sont pas supportées.

Exemple pour ingres_query()

```
<?php
ingres_connect($database, $user, $password);
ingres_query("select * from table");
while ($row = ingres_fetch_row()) {
    echo $row[1];
    echo $row[2];
}
?>
```

Voir aussi [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#), [ingres_fetch_row\(\)](#), [ingres_commit\(\)](#), [ingres_rollback\(\)](#) et [ingres_autocommit\(\)](#).

10.35.5 [ingres_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ingres_num_rows](#) (resource *link*)

[PHP 4 >= 4.0.2]

Pour les requêtes DELETE, INSERT, UPDATE [ingres_num_rows\(\)](#) retourne le nombre de lignes (tuples) affectées par la requête. Pour les autres requêtes, [ingres_num_rows\(\)](#) retourne le nombre de lignes du résultat de la requête.

Note : Cette fonction est conçue principalement pour obtenir le nombre de tuples modifiés dans la base de données. Si cette fonction est appelée avant d'utiliser [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) ou [ingres_fetch_row\(\)](#), le serveur efface les données du résultat et le script ne pourra plus les obtenir. Il faut dans ce cas récupérer les données du résultat en utilisant une de ces 3 fonctions dans une boucle jusqu'à ce qu'elle renvoie FALSE, ce qui indique qu'il n'y a plus de résultats à récupérer.

Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.6 [ingres_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ingres_num_fields](#) (resource *link*)

[PHP 4 >= 4.0.2]

[ingres_num_fields\(\)](#) retourne le nombre de champs du résultat renvoyé par le serveur Ingres après un appel à [ingres_query\(\)](#).

Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.7 [ingres_field_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ingres_field_name](#) (int *index* , resource *link*)

[PHP 4 >= 4.0.2]

[ingres_field_name\(\)](#) retourne le nom d'un champ dans le résultat d'une requête, ou FALSE en cas d'échec.

index est le numéro du champ et doit être compris entre 1 et la valeur donnée par [ingres_num_fields\(\)](#).

Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.8 [ingres_field_type](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ingres_field_type](#) (int *index* , resource *link*)

[PHP 4 >= 4.0.2]

[ingres_field_type\(\)](#) retourne le type d'un champ dans le résultat d'une requête, ou FALSE en cas d'échec.

Exemples de types renvoyés par cette fonction : "IIAPI_BYTE_TYPE", "IIAPI_CHA_TYPE", "IIAPI_DTE_TYPE", "IIAPI_FLT_TYPE", "IIAPI_INT_TYPE", "IIAPI_VCH_TYPE". Certains de ces types correspondent à plus d'un type SQL, selon la taille du champ (voir [ingres_field_length\(\)](#)). Par exemple "IIAPI_FLT_TYPE" peut être un float4 ou un float8. Pour plus d'informations, voir le Guide de l'utilisateur

d'Ingres/OpenAPI – Annexe C.

index est le numéro du champ et doit être compris entre 1 et la valeur donnée par [ingres_num_fields\(\)](#).
Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.9 ingres_field_nullable

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ingres_field_nullable](#) (int *index* , resource *link*)
[PHP 4 >= 4.0.2]

[ingres_field_nullable\(\)](#) retourne TRUE si le champ peut recevoir la valeur NULL et FALSE dans le cas contraire.

index est le numéro du champ et doit être compris entre 1 et la valeur donnée par [ingres_num_fields\(\)](#).
Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.10 ingres_field_length

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ingres_field_length](#) (int *index* , resource *link*)
[PHP 4 >= 4.0.2]

[ingres_field_length\(\)](#) retourne la taille d'un champ. Il s'agit du nombre d'octets utilisés par le serveur pour stocker ce champ. Pour plus d'informations, voir le Guide de l'utilisateur d'Ingres/OpenAPI – Annexe C.

index est le numéro du champ et doit être compris entre 1 et la valeur donnée par [ingres_num_fields\(\)](#).
Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.11 ingres_field_precision

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ingres_field_precision](#) (int *index* , resource *link*)
[PHP 4 >= 4.0.2]

[ingres_field_precision\(\)](#) retourne la précision d'un champ. Cette valeur est utilisée uniquement pour les types de données SQL décimal, float et money. Pour plus d'informations, voir le Guide de l'utilisateur d'Ingres/OpenAPI – Annexe C.

index est le numéro du champ et doit être compris entre 1 et la valeur donnée par [ingres_num_fields\(\)](#).
Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.12 ingres_field_scale

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ingres_field_scale](#) (int *index* , resource *link*)
[PHP 4 >= 4.0.2]

[ingres_field_scale\(\)](#) retourne l'échelle (scale) d'un champ. Cette valeur n'est utilisée que pour le type de données SQL decimal. Pour plus d'informations, voir le Guide de l'utilisateur d'Ingres/OpenAPI – Annexe C.

index est le numéro du champ et doit être compris entre 1 et la valeur donnée par [ingres_num_fields\(\)](#).
Voir aussi [ingres_query\(\)](#), [ingres_fetch_array\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.13 ingres fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ingres_fetch_array](#) (int *result_type* , resource *link*)

[PHP 4 >= 4.0.2]

[ingres_fetch_array\(\)](#) renvoie un tableau correspondant à la ligne récupérée, ou FALSE s'il n'y a plus de ligne à récupérer.

Cette fonction est une version améliorée de [ingres_fetch_row\(\)](#). En plus de stocker les données dans un tableau à indices numériques, elle peut aussi les enregistrer dans un tableau associatif, en utilisant les noms des champs comme indices.

Si plusieurs colonnes ont le même nom, la dernière colonne aura la priorité. Pour accéder aux autres colonnes du même nom, vous devez utiliser l'index numérique, ou faire un alias pour chaque colonne.

```
<?php
ingres_query(select t1.fl as foo t2.fl as bar from t1, t2);
$result = ingres_fetch_array();
$foo = $result["foo"];
$bar = $result["bar"];
?>
```

result_type peut valoir II_NUM pour un tableau à indices numériques, II_ASSOC pour un tableau associatif, ou II_BOTH (défaut) pour un tableau mixte (accessible selon les 2 méthodes).

Du point de vue de la rapidité, cette fonction est identique à [ingres_fetch_object\(\)](#), et presque aussi rapide que [ingres_fetch_row\(\)](#) (la différence est insignifiante).

Exemple pour ingres_fetch_array()

```
<?php
ingres_connect($database, $user, $password);
ingres_query("select * from table");
while ($row = ingres_fetch_array()) {
    echo $row["user_id"]; # utilisation du tableau associatif
    echo $row["fullname"];
    echo $row[1];        # utilisation du tableau à indices numériques
    echo $row[2];
}
?>
```

Voir aussi [ingres_query\(\)](#), [ingres_num_fields\(\)](#), [ingres_field_name\(\)](#), [ingres_fetch_object\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.14 ingres fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ingres_fetch_row](#) (resource *link*)

[PHP 4 >= 4.0.2]

[ingres_fetch_row\(\)](#) renvoie un tableau correspondant à la ligne récupérée, ou FALSE s'il n'y a plus de ligne à récupérer. la ligne est stockée dans un tableau à indices numériques, le premier champ étant à l'indice 1.

Les appels successifs à [ingres_fetch_row\(\)](#) retournent les lignes suivantes du résultat, ou FALSE s'il n'y a plus de lignes.

Exemple pour [ingres_fetch_row\(\)](#)

```
<?php
ingres_connect($database, $user, $password);
ingres_query ("select * from table");
while ($row = ingres_fetch_row()) {
echo $row[1];
echo $row[2];
}
?>
```

Voir aussi [ingres_num_fields\(\)](#), [ingres_query\(\)](#), [ingres_fetch_array\(\)](#) et [ingres_fetch_object\(\)](#).

10.35.15 ingres_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [ingres_fetch_object](#) (int *result_type* , resource *link*)

[PHP 4 >= 4.0.2]

[ingres_fetch_object\(\)](#) renvoie un objet correspondant à la ligne (tuple) récupérée, ou FALSE s'il n'y a plus de ligne à récupérer.

[ingres_fetch_object\(\)](#) est similaire à [ingres_fetch_array\(\)](#), avec une différence : la valeur de retour est un objet et non un tableau. Indirectement, cela signifie qu'il n'est possible d'accéder aux données qu'avec les noms des champs, et pas avec leur numéro (les nombres ne sont pas des noms de propriété valide).

Le paramètre optionnel *result_type* est une constante qui peut prendre les valeurs II_ASSOC, II_NUM, et II_BOTH (par défaut).

Du point de vue de la rapidité, cette fonction est identique à [ingres_fetch_array\(\)](#), et presque aussi rapide que [ingres_fetch_row\(\)](#) (la différence est insignifiante).

Exemple pour [ingres_fetch_object\(\)](#)

```
<?php
ingres_connect($database, $user, $password);
ingres_query("select * from table");
while ($row = ingres_fetch_object()) {
echo $row->user_id;
echo $row->fullname;
}
?>
```

Voir aussi [ingres_query\(\)](#), [ingres_num_fields\(\)](#), [ingres_field_name\(\)](#), [ingres_fetch_array\(\)](#) et [ingres_fetch_row\(\)](#).

10.35.16 ingres_rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ingres_rollback](#) (resource *link*)

[PHP 4 >= 4.0.2]

[ingres_rollback\(\)](#) annule (roll back) la transaction ouverte, ce qui annule les modifications faites sur la base de données au cours de cette transaction.

Ceci ferme la transaction. une nouvelle transaction peut être ouverte en envoyant une requête à l'aide de [ingres_query\(\)](#).

Voir aussi [ingres_query\(\)](#), [ingres_commit\(\)](#) et [ingres_autocommit\(\)](#).

10.35.17 ingres_commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ingres_commit](#) (resource *link*)

[PHP 4 >= 4.0.2]

[ingres_commit\(\)](#) valide la transaction ouverte, ce qui rend permanentes toutes les modifications faites sur la base de données au cours de cette transaction

Ceci ferme la transaction. une nouvelle transaction peut être ouverte en envoyant une requête à l'aide de [ingres_query\(\)](#).

Vous pouvez aussi faire en sorte que le serveur valide automatiquement les changements après chaque requête en appelant [ingres_autocommit\(\)](#) avant l'ouverture d'une transaction.

Voir aussi [ingres_query\(\)](#), [ingres_rollback\(\)](#) et [ingres_autocommit\(\)](#).

10.35.18 ingres_autocommit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [ingres_autocommit](#) (resource *link*)

[PHP 4 >= 4.0.2]

[ingres_autocommit\(\)](#) est appelée juste avant l'ouverture d'une transaction (avant le premier appel à [ingres_query\(\)](#) ou juste après un appel à [ingres_rollback\(\)](#) ou [ingres_autocommit\(\)](#)) pour changer l'état du mode "autocommit" (ce mode fonctionne comme une bascule). Lorsque le script débute le mode autocommit est toujours désactivé.

Lorsque le mode autocommit est activé, chaque requête est automatiquement commise (validée) par le serveur, comme si [ingres_commit\(\)](#) était appelée après chaque appel à [ingres_query\(\)](#).

Voir aussi [ingres_query\(\)](#), [ingres_rollback\(\)](#) et [ingres_commit\(\)](#).

10.36 LDAP

[\[Notes en ligne\]](#)

10.36.1 Introduction à LDAP

[\[Notes en ligne\]](#)

LDAP est l'acronyme de Lightweight Directory Access Protocol, c'est à dire Protocole Léger d'Accès aux Dossiers. C'est un protocole utilisé pour accéder à des "serveurs de dossiers", des serveurs qui gèrent les informations de manière hiérarchique.

Le concept est similaire à la structure de votre disque dur, hormis le fait que la racine s'appelle ici : "The world" (le monde), et que les dossiers du premier niveau sont assimilés à des pays. Les niveaux inférieurs de la structure contiennent des entrées de sociétés, d'organisations ou de lieux tandis que les niveaux encore

inférieurs sont des gens, voire des équipements ou des documents.

Pour accéder à un fichier sur votre disque, vous devez utiliser la syntaxe suivante :

```
/usr/local/myapp/docs
```

Le slash indique une division de la référence, et la séquence est lue de gauche à droite.

Avec tous les détails, une référence LDAP s'appelle un "nom distingué" ("distinguished name"), appelé aussi "nd" ("dn" en anglais). Par exemple :

```
cn=Jean Dupont,ou=Comptes,o=Ma Société,c=Fr
```

La virgule marque une division de la référence, et la séquence est lue de droite à gauche. Vous pouvez la lire comme ceci :

```
country = Fr
organization = Ma Société
organizationalUnit = Comptes
commonName = Jean Dupont
```

De la même façon qu'il n'y a pas de règle universelle d'organisation d'un disque dur, un serveur de dossier peut supporter n'importe quelle structure du moment qu'elle a un sens pour ce qu'on en fait. Cependant, il existe quelques conventions : il est impossible d'écrire un code d'accès à un dossier sans en connaître sa structure, de la même façon que vous ne pouvez pas utiliser une base de données sans en connaître les tables.

10.36.2 Exemple complet

[\[Notes en ligne\]](#)

Recupérer toutes les entrées dont le nom commence par "S" dans un serveur, et afficher le nom et l'adresse email.

Recherche LDAP

```
<?php
// Structure d'une commande simple :
// connexion, lien, recherche, interpretation de la recherche
// résultat, déconnexion
echo "<?h3>LDAP query test<?/h3>";
echo "Connexion ...";
$ds=ldap_connect("localhost"); // Doit être un serveur LDAP valide!
echo "Résultat de la connexion : ".$ds."<?p>";
if ($ds) {
echo "Lien ...";
    $r=ldap_bind($ds);          // Ceci est un lien "anonymous", typiquement
                                // en lecture seule. En cas d'accès, affiche
                                // " Lien résultat est"
echo "Lien résultat est ".$r."<?p>";
echo "Recherche de (sn=S*) ...";
    // Recherche dans les noms
    $sr=ldap_search($ds,"o=Ma Société, c=Fr", "sn=S*");
echo "Résultat : ".$sr."<?p>";
echo "Nombre d'entrée retournée : ".ldap_count_entries($ds,$sr)."<?p>";
echo "Lecture des entrées...<?p>";
    $info = ldap_get_entries($ds, $sr);
echo "Data for ".$info["count"]." items returned:<?p>";
for ($i=0; $i<?$info["count"]; $i++) {
echo "dn vaut : ". $info[$i]["dn"] ."<?br>";
```



```

echo "première entrée cn vaut : ". $info[$i]["cn"][0] . "<br>";
echo "première email vaut: ". $info[$i]["mail"][0] . "<p>";
}
echo "Déconnexion ";
ldap_close($ds);
} else {
echo "<h4>Impossible de se connecter à un serveur LDAP </h4>";
}
?>

```

10.36.2.1 Utilisation des fonctions PHP LDAP

[\[Notes en ligne\]](#)

Il faut d'abord que les bibliothèques client LDAP soient compilées avec PHP. Vous pouvez vous procurer ces bibliothèques University of Michigan (ldap-3.3 package) ou chez Netscape (Netscape Directory SDK). Avant d'utiliser les fonctions LDAP il faut savoir :

- Le nom ou l'adresse du serveur à utiliser
- Le "nd" dans le serveur (la partie du monde qui est sur ce serveur, ce qui peut correspondre à "o=Ma société,c=Fr")
- Eventuellement, un mot de passe pour accéder au serveur (de nombreux serveur fournissent un accès en lien anonymes ("anonymous bind") mais requièrent un mot de passe pour tout le reste).

Une séquence habituelle LDAP suivra le schéma suivant :

```

ldap_connect()      // établit une connexion à un serveur
|
ldap_bind()         // nom de compte "login" ou anonyme
|
    exécution de commandes sur le serveur, comme un listage, ou
une modification de données avec affichage
|
ldap_close()        // "déconnexion"

```

10.36.2.2 Plus d'informations

[\[Notes en ligne\]](#)

Vous pouvez en apprendre encore plus, mais en anglais, aux adresses suivantes :

- [Netscape](#)
- [University of Michigan](#)
- [OpenLDAP Project](#)
- [LDAP World](#)

Le SDK Netscape contient un guide du programmeur au format HTML particulièrement pratique (en anglais).

10.36.3 ldap_add

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_add](#)(int *link_identifieur*, string *dn*, array *entry*)

[PHP 3, PHP 4]

[ldap_add\(\)](#) retourne TRUE en cas de succès, ou FALSE en cas d'erreur.

[ldap_add\(\)](#) sert à ajouter une entrée dans un dossier LDAP. Le ND de l'entrée sera ajouté à la *dn* du dossier spécifié. Le tableau *entry* spécifie les informations de la nouvelle entrée. Les valeurs de l'entrée sont indexées dans les attributs de l'entrée. Si un attribut a de multiples valeurs, elles seront indexées dans un tableau, à partir de l'index 0.

```
entree["attribut1"] = valeur
entree["attribut2"][0] = valeur1
entree["attribut2"][1] = valeur2
```

Exemple complet avec lien authentifié

```
<?php
$ds=ldap_connect("localhost"); // On suppose que le serveur LDAP est sur cet hôte
if ($ds) {
    // liaison avec le nd approprié, pour avoir un accès en modification
    $r=ldap_bind($ds,"cn=root, o=Ma Société, c=Fr", "secret");
    // preparation des données
    $info["cn"]="John Jones";
    $info["sn"]="Jones";
    $info["mail"]="jonj@here.and.now";
    $info["objectclass"]="person";
    // Ajout des données dans le dossier
    $r=ldap_add($ds, "cn=John Jones, o=My Company, c=US", $info);
    ldap_close($ds);
} else {
    echo "Impossible de se connecter au serveur LDAP ";
}
?>
```

10.36.4 ldap_bind

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_bind](#)(int *link_identifieur*, string *bind_rdn*, string *bind_password*)

[PHP 3, PHP 4]

[ldap_bind\(\)](#) lie un serveur LDAP avec le RDN *bind_rdn* et mot de passe *bind_password*.

[ldap_bind\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_bind\(\)](#) effectue une opération de liaison sur le serveur. *bind_rdn* et *bind_password* sont optionnels. Si ils manquent, la liaison se fera en mode anonyme.

10.36.5 ldap_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_close](#)(int *link_identifieur*)

[PHP 3, PHP 4]

[ldap_close\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_close\(\)](#) ferme le lien au serveur LDAP associé à l'identifiant *link_identifieur*.

Cet appel est identique à [ldap_unbind\(\)](#), en interne. Les API LDAP utilisent la fonction [ldap_unbind\(\)](#) : il est probablement mieux que vous utilisiez cette fonction là plutôt que [ldap_close\(\)](#).

10.36.6 ldap_compare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_compare](#) (int *link_identifiant*, string *dn*, string *attribute*, string *value*)
[PHP 4 >= 4.0.2]

[ldap_compare\(\)](#) retourner TRUE si *value* un fichier correspond à la recherche; retourne -1 si une erreur survient.

[ldap_compare\(\)](#) sert à comparer la valeur *value* de l'attribut *attribute* aux valeurs du même attribut dans l'annuaire LDAP *dn*.

L'exemple suivant montre comment vérifier qu'un mot de passe correspond bien à celui qui est stocké dans l'annuaire.

Vérification d'un mot de passe avec LDAP

```
<?php
$ds=ldap_connect("localhost"); // on suppose que le serveur LDAP est sur le serveur local
if ($ds) {
    // liaison
    if(ldap_bind($ds)) {
        // prépare les données
        $dn = "cn=Matti Meikku, ou=My Unit, o=My Company, c=FI";
        $value = "secretpassword";
        $attr = "password";
        // compare les valeurs
        $r=ldap_compare($ds, $dn, $attr, $value);
        if ($r === -1) {
            echo "Erreur: ".ldap_error($ds);
        } elseif ($r === TRUE) {
            echo "Mot de passe correct.";
        } elseif ($r === FALSE) {
            echo "Mot de passe erroné!";
        }
    } else {
        echo "Connexion impossible.";
    }
    ldap_close($ds);
} else {
    echo "Impossible de se connecter au serveur LDAP.";
}
?>
```

Note : [ldap_compare\(\)](#) ne peut pas comparer des données binaires.

Note : Cette fonction a été ajoutée en 4.0.2.

10.36.7 ldap_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_connect](#) (string *hostname*, int *port*)
[PHP 3, PHP 4]

[ldap_connect\(\)](#) retourne un identifiant positif de serveur LDAP en cas de succès, ou bien FALSE en cas d'erreur.

[ldap_connect\(\)](#) établit une connexion avec un serveur. Le serveur LDAP situé sur l'hôte *hostname* et *port*. Les deux arguments sont optionnels. Sans argument, l'identifiant de la dernière connexion ouverte sera retournée. Si seul *hostname* est spécifié, le port par défaut est 389.

Si vous utilisez OpenLDAP 2.x.x, vous pouvez spécifier une URL au lieu d'un nom d'hôte. Pour utiliser LDAP avec SSL, compilez OpenLDAP 2.x.x avec le support SSL, configurez PHP avec SSL, et utilisez `ldaps://hostname/` comme nom d'hôte. Le paramètre de port *port* n'est pas utile lorsqu'utilisé avec des URL. Le support des URL et SSL a été ajouté dans PHP 4.0.4.

[10.36.8 ldap_count_entries](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_count_entries](#)(int *link_identifiant*, int *result_identifiant*)
[PHP 3, PHP 4]

[ldap_count_entries\(\)](#) retourne le nombre d'entrées en cas de succès, et FALSE sinon.

[ldap_count_entries\(\)](#) retourne le nombre d'entrées placées dans le résultat par les recherches précédentes. *result_identifiant* identifie un résultat LDAP interne.

[10.36.9 ldap_delete](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_delete](#)(int *link_identifiant*, string *dn*)
[PHP 3, PHP 4]

[ldap_delete\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_delete\(\)](#) efface une entrée dans un dossier LDAP spécifié par le nd *dn*.

[10.36.10 ldap_dn2ufn](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_dn2ufn](#)(string *dn*)
[PHP 3, PHP 4]

[ldap_dn2ufn\(\)](#) sert à mettre le nd *dn* dans un format plus agréable, notamment en supprimant les noms des types.

[10.36.11 ldap_err2str](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_err2str](#)(int *errno*)
[PHP 3 >= 3.0.13, PHP 4 >= 4.0RC2]

[ldap_err2str\(\)](#) retourne un message d'erreur.

[ldap_err2str\(\)](#) retourne le message d'erreur lié au numér d'erreur *errno*. Même si les numéros d'erreur LDAP sont standardisés, différentes bibliothèques retournent différents messages, ou parfois, des messages en langue locale. Ne vous fiez pas au message d'erreur, mais bien au numéro d'erreur.

Voir aussi [ldap_errno\(\)](#) et [ldap_error\(\)](#).

Enumérer tous les messages d'erreur LDAP

```
<?php
for($i=0; $i<100; $i++) {
printf("Error $i: %s<br>\n", ldap_str2err($i));
}
?>
```

10.36.12 ldap_errno

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_errno](#) (int *link_id*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0RC2]

[ldap_errno\(\)](#) retourne le numéro d'erreur LDAP généré par la dernière commande.

[ldap_errno\(\)](#) retourne le numéro d'erreur standard, généré par la dernière commande LDAP, pour la connexion *link_id*. Ce numéro peut être converti en message textuel avec [ldap_err2str\(\)](#).

A moins que vous n'abaissiez suffisamment le niveau d'erreur dans ``php.ini'` (ou ``php3.ini'`), ou que vous ne préfixiez vos commandes LDAP avec (at) pour supprimer les affichages, les erreurs LDAP s'afficheront aussi dans le code PHP.

Générer et intercepter une erreur

```
<?php
// Cet exemple contient une erreur, que nous allons intercepter.
$lid = ldap_connect("localhost");
$bind = ldap_bind($lid);
// Erreur de syntaxe dans l'expression du filtre (errno 87),
// ce doit être "objectclass=*"
$res = @ldap_search($lid, "o=Myorg, c=DE", "objectclass");
if (!$res) {
printf("LDAP-Errno: %s<br>\n", ldap_errno($lid));
printf("LDAP-Error: %s<br>\n", ldap_error($lid));
die("Argh!<br>\n");
}
$info = ldap_get_entries($lid, $res);
printf("%d entrées trouvées.<br>\n", $info["count"]);
?>
```

Voir aussi [ldap_err2str\(\)](#) et [ldap_error\(\)](#).

10.36.13 ldap_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_error](#) (int *link_id*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0RC2]

[ldap_error\(\)](#) retourne un message d'erreur.

[ldap_error\(\)](#) retourne le message d'erreur lié à la connexion *link_id*. Même si les numéros d'erreur LDAP sont standardisés, différentes bibliothèques retournent différents messages, ou parfois, des messages en langue locale.

Ne vous fiez pas au message d'erreur, mais bien au numéro d'erreur.

A moins que vous n'abaissiez suffisamment le niveau d'erreur dans ``php.ini'` (ou ``php3.ini'`), ou que vous ne préfixiez vos commandes LDAP avec `ldap_` pour supprimer les affichages, les erreurs LDAP s'afficheront aussi dans le code PHP.

Voir aussi [ldap_err2str\(\)](#) et [ldap_errno\(\)](#).

10.36.14 ldap_explode_dn

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_explode_dn](#) (string *dn*, int *with_attrib*)

[PHP 3, PHP 4]

[ldap_explode_dn\(\)](#) sert à scinder le nd *dn* retourné par [ldap_get_dn\(\)](#) en plusieurs composants. Chaque composant est reconnu sous le nom Nom Distinct Relatif (ou RDN, en anglais). [ldap_explode_dn\(\)](#) retourne un tableau qui contient ces composants. *with_attrib* sert à préciser si le RDN est retourné avec ses attributs, ou seul. Pour obtenir le RDN et ses attributs, mettez *with_attrib* à 0 et pour n'avoir que les valeurs, mettez le à 1.

10.36.15 ldap_first_attribute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_first_attribute](#) (int *link_identifiant*, int *result_entry_identifiant*, int *ber_identifiant*)

[PHP 3, PHP 4]

[ldap_first_attribute\(\)](#) retourne le premier attribut en cas de succès, et FALSE sinon.

Le comportement est similaire pour les entrées. Les attributs sont lus séquentiellement dans une entrée particulière. [ldap_first_attribute\(\)](#) retourne le premier attribut de l'entrée désignée par l'identifiant d'entrée.

Les attributs suivants sont accessibles avec [ldap_next_attribute\(\)](#). *ber_identifiant* est un identifiant de pointeur de mémoire interne. Il est passé par référence. Le même identifiant *ber_identifiant* est passé à

[ldap_next_attribute\(\)](#), qui modifie ce pointeur.

Voir aussi [ldap_get_attributes\(\)](#).

10.36.16 ldap_first_entry

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_first_entry](#) (int *link_identifiant*, int *result_identifiant*)

[PHP 3, PHP 4]

[ldap_first_entry\(\)](#) retourne un identifiant sur la première entrée en cas de succès, et FALSE sinon.

Les entrées d'un résultat sont lues séquentiellement, en utilisant [ldap_first_entry\(\)](#) et [ldap_next_entry\(\)](#).

[ldap_first_entry\(\)](#) retourne l'identifiant de la première entrée du résultat. Cet identifiant sera fourni à

[ldap_next_entry\(\)](#) pour accéder à la prochaine entrée.

Voir aussi [ldap_get_entries\(\)](#).

10.36.17 ldap_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_free_result](#) (int *result_identifieur*)

[PHP 3, PHP 4]

[ldap_free_result\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_free_result\(\)](#) libère la mémoire allouée en interne pour enregistrer le résultat pointé par *result_identifieur*.

A la fin de chaque script, la mémoire sera de toute manière libérée.

Généralement, il n'y a pas besoin de libérer la mémoire, et le mécanisme automatique de fin de script est suffisant. Cependant, dans les cas où le script effectue plusieurs recherches successives, où que les résultats retournés sont très grands, [ldap_free_result\(\)](#) permet de réduire la consommation de mémoire.

10.36.18 ldap_get_attributes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_attributes](#) (int *link_identifieur*, int *result_entry_identifieur*)

[PHP 3, PHP 4]

[ldap_get_attributes\(\)](#) retourne un tableau multi-dimensionnel en cas de succès, et FALSE sinon.

[ldap_get_attributes\(\)](#) sert à simplifier la lecture des attributs et des valeurs d'une entrée dans un résultat. Le résultat est un tableau multi-dimensionnel, avec les attributs en clé, et les valeurs des attributs en valeurs.

Une fois que vous avez repéré une entrée dans un dossier, vous pouvez lire les informations de cette entrée avec cette fonction. Vous pouvez utiliser cette fonction pour créer une application qui se déplace dans les dossiers, sans en connaître la structure au préalable. Dans de nombreux cas, vous ne chercherez qu'un attribut particulier (le email, par exemple) et vous ne vous intéresserez pas aux autres valeurs.

Affichage de la liste des attributs d'une entrée

```

<?php
// $ds est l'identifiant de lien pour ce dossier
// $sr est un résultat de recherche valide, obtenu lors d'une recherche
// précédente
$entry = ldap_first_entry($ds, $sr);
$attrs = ldap_get_attributes($ds, $entry);
echo $attrs["count"]." Attributs dans cette entrée:<p>";
for ($i=0; $i<$attrs["count"]; $i++)
echo $attrs[$i]."<br>";
?>

```

Voir aussi [ldap_first_attribute\(\)](#) et [ldap_next_attribute\(\)](#).

10.36.19 ldap_get_dn

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_get_dn](#) (int *link_identifieur*, int *result_entry_identifieur*)

[PHP 3, PHP 4]

[ldap_get_dn\(\)](#) retourne le DN de l'entrée en cas de succès, et FALSE sinon.

[ldap_get_dn\(\)](#) sert à obtenir le ND d'une entrée d'un résultat.

10.36.20 ldap_get_entries

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_entries](#) (int *link_identifieur*, int *result_identifieur*)
[PHP 3, PHP 4]

[ldap_get_entries\(\)](#) retourne un tableau multi-dimensionnel en cas de succès, et FALSE sinon.

[ldap_get_entries\(\)](#) sert à simplifier la lecture d'un résultat à plusieurs entrées. Toutes les informations sont retournées sous la forme d'un tableau multi-dimensionnel. La structure de ce tableau est la suivante : Les attributs servent d'index et sont mis en minuscules (les attributs sont insensibles à la casse sur les serveurs, mais peuvent ne pas l'être quand ils sont utilisés comme index)

```

résultat ["compte"] = nombre d'entrées du résultat
résultat [0] : correspond aux détails de la première entrée :
résultat [i]["nd"] = ND de la i-ième entrée
résultat [i]["compte"] = nombre d'attributs de la i-ième entrée
résultat [i][j] = j-ième attribut de la i-ième entrée
résultat [i]["attribut"]["count"] = nombre de valeur pour l'attribut
résultat [i]["attribut"][j] = j-ième valeur de l'attribut

```

Voir aussi [ldap_first_entry\(\)](#) et [ldap_next_entry\(\)](#).

10.36.21 ldap_get_option

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [ldap_get_option](#) (int *link_identifieur*, int *option*, mixed *retval*)

[ldap_get_option\(\)](#) remplace la valeur courante de l'option *option* par *retval*, et retourne TRUE en cas de succès, FALSE sinon.

Le paramètre *option* peut prendre l'une des valeurs : LDAP_OPT_DEREF, LDAP_OPT_SIZELIMIT, LDAP_OPT_TIMELIMIT, LDAP_OPT_PROTOCOL_VERSION, LDAP_OPT_ERROR_NUMBER, LDAP_OPT_REFERRALS, LDAP_OPT_RESTART, LDAP_OPT_HOST_NAME, LDAP_OPT_ERROR_STRING, LDAP_OPT_MATCHED_DN. Elles sont décrites dans

[draft-ietf-ldapext-ldap-c-api-xx.txt](#)

[ldap_get_option\(\)](#) n'est disponible que si vous utilisez OpenLDAP 2.x.x ou Netscape Directory SDK x.x. Elle a été ajoutée dans PHP 4.0.4.

Vérification de la version du protocole

```

<?php
// $ds est un identifiant valide d'annuaire
if (ldap_get_option($ds, LDAP_OPT_PROTOCOL_VERSION, $version))
echo "La version du protocole est $version";
else
echo "Impossible de lire la version du protocole";
?>

```

Voir aussi [ldap_set_option\(\)](#).

10.36.22 ldap_get_values

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_values](#) (int *link_identifieur*, int *result_entry_identifieur*, string *attribut*)
[PHP 3, PHP 4]

[ldap_get_option\(\)](#) retourne un tableau de valeurs en cas de succès, et FALSE sinon.

[ldap_get_values\(\)](#) sert à lire toutes les valeurs d'un attribut dans une entrée. L'entrée est référencée par *result_entry_identifieur*. Le nombre de valeurs peut être trouvé à l'index "count" dans le résultat. Les valeurs sont accessibles par un index entier, qui commence à 0.

[ldap_get_values\(\)](#) nécessite un pointeur de résultat *result_entry_identifieur*, ce qui implique qu'il ait été précédé d'une recherche sur le serveur, et de l'obtention d'une entrée.

Votre application pourra utiliser des noms d'attributs en dur dans le code, ou bien, utiliser la fonction

[ldap_get_attributes\(\)](#) pour y accéder dynamiquement.

LDAP autorise plus d'une entrée par attribut, ce qui permet, par exemple, d'étiqueter tous les adresses email d'un utilisateur avec l'attribut "mail"

```
return_value["count"] = nombre de valeurs de l'attribut
return_value[0] = première valeur de l'attribut
return_value[i] = n-ième valeur de l'attribut
```

Liste toutes les valeurs avec l'attribut "mail"

```
<?php
// $ds est l'identifiant de lien pour ce dossier
// $sr est un résultat de recherche valide, obtenu lors d'une recherche
// précédente
// $entry est un identifiant valide d'entrée
$values = ldap_get_values($ds, $entry, "mail");
echo $values["count"]." Adresse email dans ce résultat.<p>";
for ($i=0; $i < $values["count"]; $i++)
echo $values[$i]."<br>";
?>
```

10.36.23 ldap_get_values_len

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ldap_get_values_len](#) (int *link_identifieur*, int *result_entry_identifieur*, string *attribut*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0RC2]

[ldap_get_values_len\(\)](#) retourne un tableau de valeurs pour l'attribut, ou bien FALSE en cas d'erreur.

[ldap_get_values_len\(\)](#) sert à lire toutes les valeurs d'un attribut d'une entrée dans un résultat. L'entrée est spécifiée par *result_entry_identifieur*. Le nombre de valeurs trouvées peut être retrouvé en indexant "count" dans le tableau résultat. On peut accéder aux valeurs individuelles avec un index numérique, commençant à 0.

[ldap_get_values_len\(\)](#) est utilisée exactement comme [ldap_get_values\(\)](#) mais elle gère les données binaires, et non pas les chaînes.

10.36.24 [ldap_list](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_list](#)(int *link_identifieur*, string *base_dn*, string *filter*, array *attributes*, int *attrsonly* , int *sizelimit* , int *timelimit* , int *deref*)

[PHP 3, PHP 4]

[ldap_list\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_list\(\)](#) effectue une recherche avec le filtre7 donnée, et limité un dossier.

LDAP_SCOPE_ONELEVEL indique que la recherche ne doit s'étendre que dans le dossier immédiatement sous le nd *base_dn*. (Equivalent à taper "ls" et obtenir la liste des fichiers et dossiers du dossier courant).

Cette appel prend un quatrième argument optionnel : un tableau contenant les attributs recherchés. Reportez vous à [ldap_search\(\)](#) pour plus de détails.

Affiche une liste d'unités organisationnelle

```
<?php
// $ds est un identifiant de connexion valide.
$basedn = "o=Ma Société, c=Fr";
$justthese = array("ou");
$sr=ldap_list($ds, $basedn, "ou=*", $justthese);
$info = ldap_get_entries($ds, $sr);
for ($i=0; $i<$info["count"]; $i++)
echo $info[$i]["ou"][0] ;
?>
```

10.36.25 [ldap_modify](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_modify](#)(int *link_identifieur*, string *dn*, array *entry*)

[PHP 3, PHP 4]

[ldap_modify\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_modify\(\)](#) sert à modifier les entrées existantes dans un dossier LDAP. La structure de l'entrée est la même que décrite dans [ldap_add\(\)](#).

10.36.26 [ldap_mod_add](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_mod_add](#)(int *link_identifieur*, string *dn*, array *entry*)

[PHP 3>= 3.0.8, PHP 4]

[ldap_mod_add\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_mod_add\(\)](#) ajoute les attributs *entry* à l'entrée *dn*. La modification s'applique au niveau local (par opposition au niveau objet). Les ajouts au niveau de l'objet sont pris en charge par [ldap_add\(\)](#).

10.36.27 ldap_mod_del

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_mod_del](#) (int *link_identifieur*, string *dn*, array *entry*)

[PHP 3>= 3.0.8, PHP 4]

[ldap_mod_del\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_mod_del\(\)](#) efface les attributs *entry* à l'entrée *dn*. La modification s'applique au niveau local (par opposition au niveau objet). Les ajouts au niveau de l'objet sont pris en charge par [ldap_delete\(\)](#).

10.36.28 ldap_mod_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_mod_replace](#) (int *link_identifieur*, string *dn*, array *entry*)

[PHP 3>= 3.0.8, PHP 4]

[ldap_mod_replace\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_mod_replace\(\)](#) remplace les attributs *entry* à l'entrée *dn*. La modification s'applique au niveau local (par opposition au niveau objet). Les ajouts au niveau de l'objet sont pris en charge par [ldap_modify\(\)](#).

10.36.29 ldap_next_attribute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ldap_next_attribute](#) (int *link_identifieur*, int *result_entry_identifieur*, int *ber_identifieur*)

[PHP 3, PHP 4]

[ldap_next_attribute\(\)](#) retourne l'attribut suivant en cas de succès, et sinon, une erreur.

[ldap_next_attribute\(\)](#) sert à lire tous les attributs d'une entrée. Le pointeur interne est géré par *ber_identifieur*.

Il est passé par référence à la fonction. Le premier appel à [ldap_next_attribute\(\)](#) est fait avec le *result_entry_identifieur* retourné par [ldap_first_attribute\(\)](#).

Voir aussi [ldap_get_attributes\(\)](#).

10.36.30 ldap_next_entry

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_next_entry](#) (int *link_identifieur*, int *result_entry_identifieur*)

[PHP 3, PHP 4]

[ldap_next_entry\(\)](#) retourne l'identifiant de l'entrée suivante, dans le résultat qui a été initialisé par [ldap_first_entry\(\)](#). Si il n'y a plus d'entrée, retourne FALSE.

[ldap_next_entry\(\)](#) sert à retrouver toutes les entrées qui sont stockées dans un résultat. Les appels successifs à [ldap_next_entry\(\)](#) retourneront les entrées une à une. Le premier appel à [ldap_next_entry\(\)](#) est fait après un appel à [ldap_first_entry\(\)](#) avec *result_entry_identifieur* retourné par [ldap_first_entry\(\)](#).

Voir aussi [ldap_get_entries\(\)](#).

10.36.31 [ldap_read](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_read](#) (int *link_identifieur*, string *base_dn*, string *filter*, array *attributes*, int *attrsonly* , int *sizelimit* , int *timelimit* , int *deref*)

[PHP 3, PHP 4]

[ldap_read\(\)](#) retourne un identifiant de résultat en cas de succès, et FALSE sinon.

[ldap_read\(\)](#) effectue une recherche avec le filter *filter* dans le dossier *base_dn* et avec l'option LDAP_SCOPE_BASE (recherche limitée au dossier, ou récursive). Cela revient à lire une entrée dans un dossier.

Les filtres vides ne sont pas autorisés. Si vous souhaitez lire toutes les informations d'un dossier, utiliser le filtre suivant : "objectClass=*". Si vous savez quel est le type des entrées dans le dossier que vous fouillez, vous pouvez aussi adapter ce filter de la façon suivante "objectClass=inetOrgPerson".

[ldap_read\(\)](#) dispose de 5 arguments optionnels. Reportez vous à [ldap_search\(\)](#).

Note : Ces paramètres optionnels ont été ajoutés à partir de la version 4.0.2: *attrsonly*, *sizelimit*, *timelimit*, *deref*.

10.36.32 [ldap_search](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_search](#) (int *link_identifieur*, string *base_dn*, string *filter*, array *attributes* , int *attrsonly* , int *sizelimit* , int *timelimit* , int *deref*)

[PHP 3, PHP 4]

[ldap_search\(\)](#) retourne un identifiant de résultat en cas de succès, et FALSE sinon.

[ldap_search\(\)](#) effectue une recherche avec le filtre *filter* dans le dossier *base_dn*, et avec l'option de récursivité LDAP_SCOPE_SUBTREE. Cela revient à rechercher dans toute la base sous le dossier *base_dn*.

Le quatrième paramètre est optionnel, et peut être ajouté pour restreindre les attributs et les valeurs retournées. Il est beaucoup plus efficace que la méthode qui consiste à lire tous les attributs et leur valeurs associées. L'utilisation de ce quatrième paramètre est encouragée.

Le quatrième paramètre est un tableau de chaînes, qui contient les attributs désirés, array("mail","sn","cn").

Notez que le nd *base_dn* est toujours retourné, quelques que soient les attributs demandés.

Notez que certains serveurs sont configurés pour limiter le nombre de résultats. Si cela arrive, le serveur indiquera qu'il n'a transféré qu'une partie du résultat. Cela arrivera aussi si le sixième paramètre *sizelimit* a été utilisé pour limiter le nombre de lignes lues.

Le cinquième paramètre *attrsonly* doit être mis à 1 si seuls les types d'attributs sont demandés. S'il est mis à 0, les types des attributs et leur valeur seront lue (comportement par défaut).

Le sixième paramètre *sizelimit* permet de limiter le nombre de lignes lues. 0 indique qu'il n'y a pas de limite.

NOTE : ce paramètre ne peut PAS annuler la configuration du serveur. Il peut seulement la réduire.

Le septième paramètre *timelimit* fixe la durée de la recherche en seconde. 0 indique qu'il n'y a pas de limite.

NOTE : ce paramètre ne peut PAS annuler la configuration du serveur. Il peut seulement la réduire.

Le huitième paramètre *deref* indique le comportement à suivre avec les alias durant une recherche. Sa valeur peut être l'une de celles-ci :

- LDAP_DEREF_NEVER – (par défaut) les alias ne sont pas déréférencés.
- LDAP_DEREF_SEARCHING – alias sont déréférencés durant la recherche mais pas lors de leur localisation.
- LDAP_DEREF_FINDING – alias sont déréférencés durant leur localisation mais pas lors de la

recherche.

- LDAP_DEREF_ALWAYS – les alias sont toujours déréférencés.

Ces paramètres optionnels ont été ajoutés à partir de la version 4.0.2: *attrsonly*, *sizelimit*, *timelimit*, *deref*. La chaîne de filtre peut être simple ou complexe. Elle utilise les opérateurs booléens au même format que celui décrit dans les documentations LDAP. (Allez voir celle de [Netscape Directory SDK](#) pour plus d'informations sur les filtres).

L'exemple suivant récupère toutes les unités organisationnelles, le nom, prénom et email, dans la société "Ma Société" où le nom et prénom contiennent la sous-chaîne \$person. Cet exemple utilise un filtre booléen pour indiquer au serveur qu'il doit rechercher des informations dans plusieurs attributs.

Recherche LDAP

```
<?php
// $ds est un identifiant valide de connexion à un serveur LDAP
// $person est tout ou une partir d'un nom
$dn = "o=Ma Société, c=Fr";
$filter="(|(sn=$person*)(givenname=$person*))";
$justthese = array( "ou", "sn", "givenname", "mail");
$sr=ldap_search($ds, $dn, $filter, $justthese);
$info = ldap_get_entries($ds, $sr);
print $info["count"]." Entrées retournées.<p>";
?>
```

10.36.33 ldap_set_option

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [ldap_set_option](#) (int *link_identifiant*, int *option*, mixed *newval*)

[ldap_set_option\(\)](#) remplace la valeur de l'option *option* par *newval*, et retourne TRUE en cas de succès, FALSE sinon.

Le paramètre *option* peut prendre l'une des valeurs suivantes : LDAP_OPT_DEREF, LDAP_OPT_SIZELIMIT, LDAP_OPT_TIMELIMIT, LDAP_OPT_PROTOCOL_VERSION, LDAP_OPT_ERROR_NUMBER, LDAP_OPT_REFERRALS, LDAP_OPT_RESTART, LDAP_OPT_HOST_NAME, LDAP_OPT_ERROR_STRING, LDAP_OPT_MATCHED_DN, LDAP_OPT_SERVER_CONTROLS, LDAP_OPT_CLIENT_CONTROLS. Pour une brève description, reportez vous au fichier [draft-ietf-ldapext-ldap-c-api-xx.txt](#).

Les options LDAP_OPT_DEREF, LDAP_OPT_SIZELIMIT, LDAP_OPT_TIMELIMIT, LDAP_OPT_PROTOCOL_VERSION et LDAP_OPT_ERROR_NUMBER ont une valeur entière, LDAP_OPT_REFERRALS et LDAP_OPT_RESTART sont des booléens, et LDAP_OPT_HOST_NAME, LDAP_OPT_ERROR_STRING et LDAP_OPT_MATCHED_DN sont des chaînes de caractères. Le premier exemple illustre leur utilisation. LDAP_OPT_SERVER_CONTROLS et LDAP_OPT_CLIENT_CONTROLS requiert une liste de contrôles, ce qui signifie que la valeur peut être un tableau de contrôles. Un contrôle est constitué d'un *oid* identifiant le contrôle, d'une valeur *value* optionnelle, et d'un flag optionnel de *criticalité*. En PHP un contrôle est un tableau ayant la clé *oid* et une valeur sous forme de chaîne, ainsi que deux éléments optionnels. Ces éléments ont pour clé *value*, sous forme de chaîne, et une clé *iscritical* contenant un booléen. Par défaut, *iscritical* vaut FALSE. Voyez aussi l'exemple ci-dessous.

Cette fonction n'est disponible que lorsque vous utilisez OpenLDAP 2.x.x OU Netscape Directory SDK x.x. Elle a été ajoutée dans PHP 4.0.4.

Modification de la version du protocole

```
<?php
// $ds est un identifiant valide d'annuaire
if (ldap_set_option($ds, LDAP_OPT_PROTOCOL_VERSION, 3))
echo "Utilisons LDAPv3";
else
echo "Impossible de changer le protocole en version 3";
?>
```

Modifier les contrôles du serveur LDAP

```
// $ds est un lien valide LDAP
// control n'est pas une valeur
$ctrl1 = array("oid" => "1.2.752.58.10.1", "iscritical" => TRUE);
// iscritical vaut par défaut FALSE $ctrl2 = array("oid" => "1.2.752.58.1.10", "value" => "magic")
// modification des deux contrôles
if (!ldap_set_option($ds, LDAP_OPT_SERVER_CONTROLS, array($ctrl1, $ctrl2)))
echo "Impossible de se connecter au serveur";
```

Voir aussi [ldap_get_option\(\)](#).

10.36.34 ldap_unbind

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ldap_unbind](#) (int *link_identifiant*)

[PHP 3, PHP 4]

[ldap_unbind\(\)](#) retourne TRUE en cas de succès, et FALSE sinon.

[ldap_unbind\(\)](#) termine la liaison avec le serveur LDAP.

10.37 Email

[\[Notes en ligne\]](#)

[mail\(\)](#) envoie du courrier électronique.

10.37.1 mail

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mail](#) (string *to*, string *subject*, string *message*, string *additional_headers*)

[PHP 3, PHP 4]

[mail\(\)](#) poste automatiquement le message *message* à destination de *to*. Les destinataires multiples doivent être séparés par des virgules.

Envoi de mail.

```
<?php
```

```
mail("rasmus@lerdorf.on.ca", "Mon Sujet", "Ligne 1\nLigne 2\nLigne 3");
?>
```

Le quatrième argument passé sera inséré à la fin de l'entête. Typiquement, cela permet d'insérer des entêtes supplémentaires. Les entêtes multiples doivent être séparés par des virgules.

Envoi de eMail avec des entêtes supplémentaires.

```
<?php
mail("nobody@aol.com", "Le sujet", $message,
    "From: webmaster@$SERVER_NAME\nReply-To: webmaster@$SERVER_NAME\nX-Mailer: PHP/" . phpversion()
);
?>
```

Vous pouvez aussi utiliser des techniques simples de concaténations de chaînes pour construire des messages complexes :

Envoi de mail complexe.

```
<?php
/* destinataire */
$recipient .= "Mary <mary@u.college.edu>" . ", " ; //remarquez les virgules
$recipient .= "Kelly <kelly@u.college.edu>" . ", " ;
$recipient .= "ronabop@php.net";
/* sujet */
$subject = "Rappel des anniversaires du mois d'aout";
/* message */
$message .= "Le mail suivant inclus une table au format ASCII\n";
$message .= "Jour \t\tMois \t\tAn\n";
$message .= "3 \t\tAou \t\t1970\n";
$message .= "17\t\tAou \t\t1973\n";
/* Vous pouvez ajouter une signature */
$message .= "--\r\n"; //Délimiteur de signature
$message .= "Rappel d'anniversaire : copyleft par public domain";
/* d'autres entêtes : errors, From cc's, bcc's, etc */
$headers .= "From: Rappel d'anniversaire <birthday@php.net>\n";
$headers .= "X-Sender: <birthday@php.net>\n";
$headers .= "X-Mailer: PHP\n"; // mailer
$headers .= "X-Priority: 1\n"; // Message urgent!
$headers .= "Return-Path: <birthday@php.net>\n"; // Re-chemin de retour pour les erreurs
$headers .= "Content-Type: text/html; charset=iso-8859-1\n" // Type MIME
$headers .= "cc:birthdayarchive@php.net\n"; // Champs CC
$headers .= "bcc:birthdaycheck@php.net, birthdaygifts@php.net\n"; // Champs BCCs
/* et hop, à la poste */
mail($recipient, $subject, $message, $headers);
?>
```

10.37.2 ezmlm_hash

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ezmlm_hash](#)(string *addr*)

[PHP 3>= 3.0.17, PHP 4 >= 4.0.2]

[ezmlm_hash\(\)](#) calcule la valeur de hash, nécessaire lors de la gestion de liste de diffusions EZMLM dans une base de données MySQL.

Calcul du hash et enregistrement d'un utilisateur

```
<?php
$user = "kris@koehntopp.de";
$hash = ezmlm_hash ($user);
$query = sprintf ("INSERT INTO sample VALUES (%s, '%s')", $hash, $user);
$db->query($query); // utilisation de l'interface PHPLIB
?>
```

10.38 Mathématiques

[\[Notes en ligne\]](#)

10.38.1 Introduction

[\[Notes en ligne\]](#)

Ces fonctions ne sont capables de manipuler que des entiers double, ou des long. Si vous avez besoin de manipuler des nombres plus grands, reportez vous aux fonctions mathématiques sur des [10.4 Nombres de grande taille](#).

10.38.1.1 Constantes mathématiques

[\[Notes en ligne\]](#)

Les valeurs suivantes sont définies comme des constantes dans PHP:

Constante	Valeur	Description
M_PI	3.14159265358979323846	Pi
M_E	2.7182818284590452354	e
M_LOG2E	1.4426950408889634074	log ₂ e
M_LOG10E	0.43429448190325182765	log ₁₀ e
M_LN2	0.69314718055994530942	log _e 2
M_LN10	2.30258509299404568402	log _e 10
M_PI_2	1.57079632679489661923	pi/2
M_PI_4	0.78539816339744830962	pi/4
M_1_PI	0.31830988618379067154	1/pi
M_2_PI	0.63661977236758134308	2/pi
M_SQRTPI	1.77245385090551602729	sqrt(pi) [4.0.2]
M_2_SQRTPI	1.12837916709551257390	2/sqrt(pi)
M_SQRT2	1.41421356237309504880	sqrt(2)
M_SQRT3	1.73205080756887729352	sqrt(3) [4.0.2]
M_SQRT1_2	0.70710678118654752440	1/sqrt(2)

M_LNPI	1.14472988584940017414	log_e(pi) [4.0.2]
M_EULER	0.57721566490153286061	Euler constant [4.0.2]

10.38.2 abs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [abs](#) (mixed *number*)
[PHP 3, PHP 4]

[abs\(\)](#) retourne la valeur absolue du nombre *number*. Si le nombre est de type float, le type retourné est float, sinon, c'est integer (entier).

10.38.3 acos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [acos](#) (float *arg*)
[PHP 3, PHP 4]

[acos\(\)](#) retourne l'arc cosinus de *arg* (*arg* en radians).
Voir aussi [asin\(\)](#) et [atan\(\)](#).

10.38.4 asin

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [asin](#) (float *arg*)
[PHP 3, PHP 4]

[asin\(\)](#) retourne l'arc sinus de *arg* (*arg* en radians).
Voir aussi [acos\(\)](#) et [atan\(\)](#).

10.38.5 atan

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [atan](#) (float *arg*)
[PHP 3, PHP 4]

[atan\(\)](#) retourne l'arc tangent de *arg* (*arg* en radians).
Voir aussi [acos\(\)](#) et [atan\(\)](#).

10.38.6 atan2

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [atan2](#) (float *y*, float *x*)
[PHP 3>= 3.0.5, PHP 4]

[atan2\(\)](#) retourne l'arc tangent de deux variables *x* et *y*. La formule est : " arc tangent (*y* / *x*) ", et les signes des arguments sont utilisés pour déterminer le quadrant du résultat.

[atan2\(\)](#) retourne un résultat en radians, entre $-\pi$ et π (inclus).

Voir aussi [acos\(\)](#) et [atan\(\)](#).

10.38.7 base_convert

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [base_convert](#) (string *number*, int *frombase*, int *tobase*)

[PHP 3 >= 3.0.6, PHP 4]

[base_convert\(\)](#) retourne une chaîne contenant l'argument *number* représenté dans la base *tobase*. La base de représentation de *number* est donnée par *frombase*. *frombase* et *tobase* doivent être compris entre 2 et 36, inclus. Les chiffres supérieurs à 10 des bases supérieures à 10 seront représentées par les lettres de a à z, avec a = 10 et z = 36.

base_convert()

```
<?php
$binary = base_convert($hexadecimal, 16, 2);
?>
```

10.38.8 bindec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [bindec](#) (string *binary_string*)

[PHP 3, PHP 4]

[bindec\(\)](#) retourne la conversion d'un nombre binaire représenté par la chaîne *binary_string* en décimal.

[bindec\(\)](#) convertit un nombre binaire en décimal. Le plus grand nombre convertible a 31 bits à 1, soit 2147483647 en décimal.

Voir aussi [decbin\(\)](#).

10.38.9 ceil

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ceil](#) (float *number*)

[PHP 3, PHP 4]

[ceil\(\)](#) retourne l'entier supérieur du nombre *number*. Utiliser [ceil\(\)](#) sur un entier ne sert à rien.

```
<?php
$x = ceil(4.25);
// ce qui donne $x=5
?>
```

NOTE: [ceil\(\)](#) sous PHP/FI 2 retournait un nombre à virgule flottante. Utilisez: \$new = (double)ceil(\$number); pour retrouver le comportement traditionnel.

Voir aussi [floor\(\)](#) et [round\(\)](#).

10.38.10 cos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [cos](#) (float *arg*)
[PHP 3, PHP 4]

[cos\(\)](#) retourne le cosinus de *arg* (*arg* en radians).
Voir aussi [sin\(\)](#) et [tan\(\)](#).

10.38.11 decbin

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [decbin](#) (int *number*)
[PHP 3, PHP 4]

[decbin\(\)](#) retourne une chaîne contenant la représentation binaire de l'entier donné en argument. Le plus grand nombre pouvant être converti est 2147483647 en décimal, ce qui donne une série de 31 uns (1).
Voir aussi [bindec\(\)](#).

10.38.12 dechex

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [dechex](#) (int *number*)
[PHP 3, PHP 4]

[dechex\(\)](#) retourne une chaîne contenant la représentation hexadécimale du nombre *number*. Le nombre le plus grand qui puisse être converti est 2147483647 en décimal, ce qui donnera "7fffffff".
Voir aussi [hexdec\(\)](#).

10.38.13 decoct

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [decoct](#) (int *number*)
[PHP 3, PHP 4]

[decoct\(\)](#) retourne une chaîne contenant la représentation octale du nombre donné *number*. Le nombre le plus grand qui puisse être converti est 2147483647 en décimal, ce qui donnera "1777777777".
Voir aussi [octdec\(\)](#).

10.38.14 deg2rad

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [deg2rad](#) (double *number*)
[PHP 3>= 3.0.4, PHP 4]

[deg2rad\(\)](#) convertit *number* de degrés en radians.

Voir aussi [rad2deg\(\)](#).

[10.38.15 exp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [exp](#)(float *arg*)

[PHP 3, PHP 4]

[exp\(\)](#) retourne l'exponentielle de *arg*, c'est à dire e élevé à la puissance *arg*.

Voir aussi [pow\(\)](#).

[10.38.16 floor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [floor](#)(float *number*)

[PHP 3, PHP 4]

[floor\(\)](#) retourne l'entier inférieur du nombre *number*. Utiliser [floor\(\)](#) sur un entier ne sert à rien.

NOTE: [floor\(\)](#) sous PHP/FI retournait un float. Utilisez: \$new = (double)floor(\$number); pour retrouver le comportement traditionnel.

Voir aussi [ceil\(\)](#) et [round\(\)](#).

[10.38.17 getrandmax](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getrandmax](#)(void)

[PHP 3, PHP 4]

[getrandmax\(\)](#) retourne la plus grande valeur aléatoire possible retournée par [rand\(\)](#).

Voir aussi [rand\(\)](#), [srand\(\)](#) [mt_rand\(\)](#), [mt_srand\(\)](#) et [mt_getrandmax\(\)](#).

[10.38.18 hexdec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [hexdec](#)(string *hex_string*)

[PHP 3, PHP 4]

[hexdec\(\)](#) retourne une chaîne contenant la représentation décimal du nombre *hex_string*. Le nombre le plus grand qui puisse être converti est 7fffffff en décimal, ce qui donne "2147483647".

Voir aussi the [dechex\(\)](#).

[10.38.19 lcg_value](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [lcg_value](#)

[PHP 4 >= 4.0b4]

[lcg_value\(\)](#) retourne un nombre pseudo-aléatoire, compris entre 0 et 1. [lcg_value\(\)](#) combine deux générateurs de congruence, de période respectives $2^{31} - 85$ et $2^{31} - 249$. La période de cette fonction est le produit de ces deux nombres premiers (soit $(2^{31} - 85) * (2^{31} - 249)$).

[10.38.20 log](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [log](#) (float *arg*)
[PHP 3, PHP 4]

[log\(\)](#) retourne le logarithme naturel de *arg*.

[10.38.21 log10](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [log10](#) (float *arg*)
[PHP 3, PHP 4]

[log10\(\)](#) retourne le logarithme en base 10 de *arg*.

[10.38.22 max](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [max](#) (mixed *arg1*, mixed *arg2*, mixed *argn*)
[PHP 3, PHP 4]

[max\(\)](#) retourne la plus grande valeur numérique parmi les valeurs passées en paramètre.

Si le premier paramètre est un tableau, [max\(\)](#) retourne la plus grande valeur de ce tableau. Si le premier paramètre est un entier, une chaîne ou un double, [max\(\)](#) requiert au moins deux paramètres, et retournera alors le plus grand d'entre eux. Le nombre d'argument est alors illimité.

Si au moins une valeur est de type double, elle seront toutes traitées comme des doubles, et un double sera retourné. Si aucune valeur n'est de type double, elles seront traitées comme des entiers, et un entier sera retourné.

[10.38.23 min](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [min](#) (mixed *arg1*, mixed *arg2*, mixed *argn*)
[PHP 3, PHP 4]

[min\(\)](#) retourne la plus petite valeur numérique parmi les valeurs passées en paramètre.

Si le premier paramètre est un tableau, [min\(\)](#) retourne la plus petite valeur de ce tableau. Si le premier paramètre est un entier, une chaîne ou un double, [min\(\)](#) requiert au moins deux paramètres, et retournera alors le plus petit d'entre eux. Le nombre d'argument est alors illimité.

Si au moins une valeur est de type double, elle seront toutes traitées comme des doubles, et un double sera retourné. Si aucune valeur n'est de type double, elles seront traitées comme des entiers, et un entier sera retourné.

10.38.24 mt_rand

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mt_rand](#)(*int min* , *int max*)

[PHP 3>= 3.0.6, PHP 4]

De nombreux générateurs de nombre aléatoires provenant de vieilles bibliothèques libcs ont des comportements douteux et sont très lents. Par défaut, PHP utilise le générateur de nombres aléatoires de libc avec la fonction [rand\(\)](#). [mt_rand\(\)](#) est une fonction de remplacement, pour cette dernière. Elle utilise un générateur de nombre aléatoire de caractéristique connue, le " Mersenne Twister ", qui va produire des nombres utilisables en cryptographie, et qui est 4 fois plus rapide que la fonction standard libc. La "Homepage of the Mersenne Twister " est <http://www.math.keio.ac.jp/~matumoto/emt.html>, une version optimisée des sources de MT est disponible à <http://www.scp.syr.edu/~marc/hawk/twister.html>. Appelé sans les arguments optionnels *min*, *max*, [mt_rand\(\)](#) retourne un nombre pseudo-aléatoire, entre 0 et RAND_MAX. Pour obtenir un nombre entre 5 et 15 (inclus), il faut utiliser [mt_rand\(5,15\)](#).

N'oubliez pas d'initialiser le générateur de nombre aléatoire avec [mt_srand\(\)](#).

Note : Dans les versions antérieure à la 3.0.7 la signification du paramètre *max* était "longueur". Pour avoir le même résultat, il faut utiliser [mt_rand\(5, 11\)](#) pour obtenir un nombre aléatoire entre 5 et 15.

Voir aussi [mt_srand\(\)](#), [mt_getrandmax\(\)](#), [srand\(\)](#), [rand\(\)](#) et [getrandmax\(\)](#).

10.38.25 mt_srand

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mt_srand](#)(int *seed*)

[PHP 3>= 3.0.6, PHP 4]

[mt_srand\(\)](#) initialise une meilleure valeur aléatoire avec *seed*.

```
<?php
// initialise avec les microsecondes depuis la dernière seconde entière
mt_srand((double)microtime()*1000000);
$randval = mt_rand();
?>
```

Voir aussi [mt_rand\(\)](#), [mt_getrandmax\(\)](#), [srand\(\)](#), [rand\(\)](#) et [getrandmax\(\)](#).

10.38.26 mt_getrandmax

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mt_getrandmax](#)(void)

[PHP 3>= 3.0.6, PHP 4]

[mt_getrandmax\(\)](#) retourne la plus grand valeur aléatoire possible que peut retourner [mt_rand\(\)](#).

Voir aussi [mt_rand\(\)](#), [mt_srand\(\)](#), [rand\(\)](#), [srand\(\)](#) et [getrandmax\(\)](#).

10.38.27 number_format

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [number_format](#) (float *number*, int *decimals*, string *dec_point*, string *thousands_sep*)

[PHP 3, PHP 4]

[number_format\(\)](#) retourne une chaîne représentant *number* formaté. [number_format\(\)](#) accepte un, deux ou 4 paramètres (mais pas trois).

Si un seul paramètre est donné, *number* sera formaté sans décimale, mais avec une virgule entre chaque série de 1000.

Avec deux paramètres, *number* sera formaté avec *decimals* décimales et un point ("."), une virgule entre chaque série de 1000.

Avec quatre paramètres, *number* sera formaté avec *decimals* décimales et *dec_point* à la place du point, une virgule entre chaque série de 1000.

10.38.28 octdec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [octdec](#) (string *octal_string*)

[PHP 3, PHP 4]

[octdec\(\)](#) retourne une chaîne contenant la représentation décimale du nombre *octal_string*. Le nombre le plus grand qui puisse être converti est 17777777777 en décimal, ce qui donnera "2147483647".

Voir aussi [decoct\(\)](#).

10.38.29 pi

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [pi](#) (void)

[PHP 3, PHP 4]

[pi\(\)](#) retourne la valeur de pi.

10.38.30 pow

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [pow](#) (float *base*, float *exp*)

[PHP 3, PHP 4]

[pow\(\)](#) retourne *base* élevé à la puissance *exp*.

Voir aussi [exp\(\)](#).

10.38.31 rad2deg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [rad2deg](#) (double *number*)

[PHP 3>= 3.0.4, PHP 4]

[rad2deg\(\)](#) convertit *number* (supposé en radians) en degrés.

Voir aussi [deg2rad\(\)](#).

[10.38.32 rand](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [rand](#) (*int min* , *int max*)

[PHP 3, PHP 4]

Appelée sans les options *min* et *max*, [rand\(\)](#) retourne un nombre pseudo-aléatoire entre 0 et RAND_MAX. Si vous voulez un nombre aléatoire entre 5 et 15 (inclus), par exemple, utilisez `rand (5, 15)`.

N'oubliez pas d'initialiser le générateur de nombres aléatoires avec [srand\(\)](#).

Note : Dans les versions antérieures à la 3.0.7 la signification du paramètre *max* était longueur. Pour avoir le même résultat, il faut utiliser `mt_rand (5, 11)` pour obtenir un nombre aléatoire entre 5 et 15.

Voir aussi [srand\(\)](#), [getrandmax\(\)](#), [mt_rand\(\)](#), [mt_srand\(\)](#) et [mt_getrandmax\(\)](#).

[10.38.33 round](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [round](#) (double *val*, int *precision*)

[PHP 3, PHP 4]

[round\(\)](#) retourne la valeur arrondie de *val* à la précision *precision* (nombre de chiffres après la virgule).

```
<?php
$foo = round( 3.4 ); // $foo == 3.0
$foo = round( 3.5 ); // $foo == 4.0
$foo = round( 3.6 ); // $foo == 4.0
?>
```

Note : Le paramètre *precision* est disponible uniquement en PHP 4.

Voir aussi [ceil\(\)](#) et [floor\(\)](#).

[10.38.34 sin](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [sin](#) (float *arg*)

[PHP 3, PHP 4]

[sin\(\)](#) retourne le sinus de *arg* (*arg* in radians).

Voir aussi [cos\(\)](#) et [tan\(\)](#).

[10.38.35 sqrt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [sqrt](#) (float *arg*)

[PHP 3, PHP 4]

[sqrt\(\)](#) retourne la racine carrée de *arg*.

10.38.36 srand

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [srand](#)(int *seed*)

[PHP 3, PHP 4]

[srand\(\)](#) initialise le générateur de nombres aléatoires avec *seed*.

```
<?php
// initialise avec les microsecondes depuis la dernière seconde entière
srand((double)microtime()*1000000);
$randval = rand();
?>
```

Voir aussi [rand\(\)](#), [getrandmax\(\)](#), [mt_rand\(\)](#), [mt_srand\(\)](#) et [mt_getrandmax\(\)](#).

10.38.37 tan

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [tan](#)(float *arg*)

[PHP 3, PHP 4]

[tan\(\)](#) retourne la tangente de *arg* (*arg* en radians).

Voir aussi [sin\(\)](#) et [cos\(\)](#).

10.39 MCAL

[\[Notes en ligne\]](#)

MCAL signifie Modular Calendar Access Library (bibliothèque calendaire modulaire).

Libmcal est une bibliothèque C de calendriers. Elle est écrite pour être très modulaire, et dispose de nombreux modules. MCAL est l'équivalent de **IMAP** pour les calendriers.

Avec mcal, un calendrier peut être ouvert comme une boîte aux lettres. Les calendriers peuvent être des fichiers locaux, ou bien être sur des serveurs ICAP distants, ou encore tout autre format supporté par la bibliothèque.

Les événements peuvent être lus, sélectionnés et enregistrés. Il y a aussi la possibilité d'ajouter des alarmes, et de placer des événements récurrents.

Avec libmcal, les serveurs centralisés peuvent être accédés et utilisés, et remplacent avantageusement tout développement spécifique de base de données.

Pour faire fonctionner cette bibliothèque, vous devez compiler PHP avec l'option `--with-mcal`. Il vous faudra alors avoir installé la bibliothèque mcal. Téléchargez la dernière version à <http://mcal.chek.com/> et compilez la, puis installez la.

Les constantes suivantes sont définies lorsque mcal est utilisée : MCAL_SUNDAY, MCAL_MONDAY, MCAL_TUESDAY, MCAL_WEDNESDAY, MCAL_THURSDAY, MCAL_FRIDAY, MCAL_SATURDAY, MCAL_RECUR_NONE, MCAL_RECUR_DAILY, MCAL_RECUR_WEEKLY,

MCAL_RECUR_MONTHLY_MDAY, MCAL_RECUR_MONTHLY_WDAY, MCAL_RECUR_YEARLY, MCAL_JANUARY, MCAL_FEBRUARY, MCAL_MARCH, MCAL_APRIL, MCAL_MAY, MCAL_JUNE, MCAL_JULY, MCAL_AUGUST, MCAL_SEPTEMBER, MCAL_OCTOBER, MCAL_NOVEMBER, et MCAL_DECEMBER. La plupart des fonctions utilisent une structure d'événement interne, qui est unique pour chaque connexion. Cela évite d'avoir à passer des objets de grande taille entre les fonctions. Il y a des accesseurs bien pratiques pour créer, initialiser et lire des objets événements.

10.39.1 mcald_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcald_open](#)(string *calendar*, string *username*, string *password*, int *options*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcald_open\(\)](#) retourne un flot MCAL en cas de succès, et FALSE en cas d'erreur.

[mcald_open\(\)](#) ouvre une connexion MCAL au serveur *calendar*. Si *options* est spécifié, passe aussi *options* à la boîte aux lettres (???). La structure interne du flot MCAL est initialisée à la connexion.

10.39.2 mcald_popen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcald_popen](#)(string *calendar*, string *username*, string *password*, int *options*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcald_popen\(\)](#) retourne un flot MCAL en cas de succès, et FALSE sinon.

[mcald_popen\(\)](#) ouvre une connexion MCAL au serveur de calendrier *calendar*. Si les options *options* sont spécifiées, elles sont aussi passées à cette boîte aux lettres. La structure interne du flot est aussi initialisée.

10.39.3 mcald_reopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcald_reopen](#)(string *calendar*, int *options*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcald_reopen\(\)](#) réouvre une connexion MCAL.

[mcald_reopen\(\)](#) réouvre une connexion MCAL avec le serveur *calendar*. Si les options *options* sont spécifiées, elles sont aussi passées à cette boîte aux lettres.

10.39.4 mcald_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcald_close](#)(int *mcald_stream*, int *flags*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcald_close\(\)](#) ferme la connexion *mcald_stream*.

10.39.5 mcal_create_calendar

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcal_create_calendar](#) (int *stream*, string *calendar*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_create_calendar\(\)](#) crée un nouveau calendrier nommé *calendar*.

10.39.6 mcal_rename_calendar

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcal_rename_calendar](#) (int *stream*, string *old_name*, string *new_name*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_rename_calendar\(\)](#) renomme le calendrier *old_name* en *new_name*.

10.39.7 mcal_delete_calendar

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcal_delete_calendar](#) (int *stream*, string *calendar*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_delete_calendar\(\)](#) efface le calendrier *calendar*.

10.39.8 mcal_fetch_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mcal_fetch_event](#) (int *mcal_stream*, int *event_id*, int *options*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_fetch_event\(\)](#) recherche un événement dans le calendrier spécifié par *id*.
Retourne un objet événement dont les attributs sont :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.
- int recur_type – type de récurrence
- int recur_interval – intervalle de récurrence
- datetime recur_enddate – date de fin de récurrence
- int recur_data – données de récurrence

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes
- int alarm – nombre de minutes avant de déclencher l'alarme

10.39.9 mcal_list_events

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcal_list_events](#) (int *mcald_stream*, object *begin_date*, object *end_date*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_list_events\(\)](#) retourne un tableau d'identifiants d'événements, compris entre deux dates.
[mcal_list_events\(\)](#) prend une date de début et une date de fin. Un tableau d'identifiants est retourné.

10.39.10 mcal_append_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_append_event](#) (int *mcald_stream*)
[PHP 4 >= 4.0RC1]

[mcal_append_event\(\)](#) enregistre l'événement global dans le calendrier MCAL *mcald_stream*.
Retourne l'uid de l'enregistrement ainsi inséré.

10.39.11 mcal_store_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_store_event](#) (int *mcald_stream*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_store_event\(\)](#) enregistre l'événement global dans le calendrier MCAL *mcald_stream*.
Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

10.39.12 mcal_delete_event

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_delete_event](#) (int *mcald_stream*, int *event_id*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_delete_event\(\)](#) efface l'événement d'identifiant *uid*.
Retourne TRUE.

10.39.13 mcal_snooze

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_snooze](#)(int *id*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_snooze\(\)](#) éteint l'alarme de l'événement identifié par l'UID *uid*.
Retourne TRUE.

10.39.14 mcal_list_alarms

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcal_list_events](#)(int *mcal_stream*, int *begin_year* , int *begin_month* , int *begin_day* , int *end_year* , int *end_month* , int *end_day*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_list_events\(\)](#) retourne un tableau d'identifiants, qui ont une alarme de prévue à la date *alarm_date*. Si seul le flot MCAL est donné, la date de début et de fin de la structure globale sera utilisée.
[mcal_list_events\(\)](#) prend une date, et retourne un tableau d'identifiants.

10.39.15 mcal_event_init

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_init](#)(int *stream*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_init\(\)](#) initialise la structure globale d'un flot. Cela remet tous les éléments de la structure à 0, ou à leur valeur par défaut.
Retourne TRUE.

10.39.16 mcal_event_set_category

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_category](#)(int *stream*, string *category*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_category\(\)](#) fixe la catégorie de la structure globale à la valeur de *category*.
Retourne TRUE.

10.39.17 mcal_event_set_title

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_title](#)(int *stream*, string *title*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_title\(\)](#) fixe le titre de la structure globale à la valeur de *title*.
Retourne TRUE.

10.39.18 mcal_event_set_description

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_description](#)(int *stream*, string *description*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_description\(\)](#) fixe la catégorie de la structure globale à la valeur de *description*.

Returns TRUE.

10.39.19 mcal_event_set_start

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_start](#)(int *stream*, int *year*, int *month*, int *day* , int *hour* , int *min* , int *sec*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_start\(\)](#) fixe la date de début de la structure globale.

Retourne TRUE.

10.39.20 mcal_event_set_end

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_end](#)(int *stream*, int *year*, int *month*, int *day* , int *hour* , int *min* , int *sec*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_end\(\)](#) fixe la date de fin de la structure globale.

Returns TRUE.

10.39.21 mcal_event_set_alarm

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_alarm](#)(int *stream*, int *alarm*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_alarm\(\)](#) fixe l'alarme de la structure globale, à un nombre de minutes avant déclenchement.

Retourne TRUE.

10.39.22 mcal_event_set_class

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_class](#)(int *stream*, int *class*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_class\(\)](#) fixe la classe de la structure globale. La classe vaut 0 pour public, et 1 pour privée.

Retourne TRUE.

10.39.23 mcal_is_leap_year

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_is_leap_year](#) (int *year*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_is_leap_year\(\)](#) retourne 1 si l'année *year* est bissextile, et 0 sinon.

10.39.24 mcal_days_in_month

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_days_in_month](#) (int *month*, int *leap year*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_days_in_month\(\)](#) retourne le nombre de jour du mois *month*, et prend en compte le fait que l'année est bissextile avec le paramètre *leap year*.

10.39.25 mcal_date_valid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_date_valid](#) (int *year*, int *month*, int *day*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_date_valid\(\)](#) retourne TRUE si la date (constituée par l'année *year*, le mois *month* et la date *day*) est valide, et FALSE sinon.

10.39.26 mcal_time_valid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_time_valid](#) (int *hour*, int *minutes*, int *seconds*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_time_valid\(\)](#) retourne TRUE si l'heure (constituée par l'heure *hour*, les minutes *minutes* et les secondes *seconds*) est une heure valide, et FALSE sinon.

10.39.27 mcal_day_of_week

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_day_of_week](#) (int *year*, int *month*, int *day*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_day_of_week\(\)](#) retourne le jour de la semaine, pour la date constituée par l'année *year*, le mois *month* et la date *day*.

10.39.28 mcal_day_of_year

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_day_of_year](#)(int *year*, int *month*, int *day*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_day_of_year\(\)](#) retourne le numéro de jour dans l'année pour la date constituée par l'année *year*, le mois *month* et la date *day*.

10.39.29 mcal_date_compare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_date_compare](#)(int *a_year*, int *a_month*, int *a_day*, int *b_year*, int *b_month*, int *b_day*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_date_compare\(\)](#) compare les deux dates données, et retourne <0, 0, >0 si a<b, a==b, a>b respectivement.

10.39.30 mcal_next_recurrence

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_next_recurrence](#)(int *stream*, int *weekstart*, array *next*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_next_recurrence\(\)](#) retourne un objet contenant la prochaine date de l'événement, ou la date de l'événement suivant la date. Retourne un objet date vide si l'événement n'a pas de réoccurrence, ou si quelquechose est invalide. Utilisez *weekstart* pour déterminer le premier jour.

10.39.31 mcal_event_set_recur_none

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_none](#)(int *stream*)

[PHP 3>= 3.0.15, PHP 4 >= 4.0RC1]

[mcal_event_set_recur_none\(\)](#) supprime la récurrence de la structure globale (event->recur_type est mis à MCAL_RECUR_NONE).

10.39.32 mcal_event_set_recur_daily

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_daily](#)(int *stream*, int *year*, int *month*, int *day*, int *interval*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_recur_daily\(\)](#) fixe la récurrence quotidienne de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.39.33 mcal_event_set_recur_weekly](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_weekly](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*, int *weekdays*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_recur_weekly\(\)](#) fixe la récurrence hebdomadaire de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.39.34 mcal_event_set_recur_monthly_mday](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_monthly_mday](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_recur_monthly_mday\(\)](#) fixe la récurrence de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.39.35 mcal_event_set_recur_monthly_wday](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_monthly_wday](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_recur_monthly_wday\(\)](#) fixe la récurrence mensuelle de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.39.36 mcal_event_set_recur_yearly](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_event_set_recur_yearly](#) (int *stream*, int *year*, int *month*, int *day*, int *interval*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_event_set_recur_yearly\(\)](#) fixe la récurrence annuelle de la structure globale, jusqu'à la date passée en paramètre. (la date de début est celle de la structure).

[10.39.37 mcal_fetch_current_stream_event](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mcal_fetch_current_stream_event](#) (int *stream*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[mcal_fetch_current_stream_event\(\)](#) retourne la structure de la date du flot courant sous la forme d'un objet,

qui contient :

- int id – ID de l'événement.
- int public – TRUE si l'événement est public, FALSE si il est privé.
- string category – Catégorie de l'événement.
- string title – Titre de l'événement.
- string description – Description de l'événement.
- int alarm – Nombre de minutes avant d'envoyer une alerte pour cet événement.
- object start – Objet contenant une date et une heure.
- object end – Objet contenant une date et une heure.
- int recur_type – type de récurrence
- int recur_interval – intervalle de récurrence
- datetime recur_enddate – date de fin de récurrence
- int recur_data – données de récurrence

Tous les objets de date et heure sont construits comme suit :

- int year – année
- int month – mois
- int mday – jour du mois
- int hour – heure
- int min – minutes
- int sec – secondes
- int alarm – nombre de minutes avant de déclencher l'alarme

10.39.38 mcal_event_add_attribute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mcal_event_add_attribute](#)(int *stream*, string *attribute*, string *value*)
[PHP 3>= 3.0.15, PHP 4 >= 4.0RC1]

[mcal_event_add_attribute\(\)](#) ajoute l'attribut *attribute* à la structure globale, avec la valeur *value*.

10.39.39 mcal_expunge

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcal_expunge](#)(int *stream*)

[mcal_expunge\(\)](#) supprime tous les événements marqués pour l'effacement.

10.40 Cryptage

[\[Notes en ligne\]](#)

Ces fonctions utilisent [mcrypt](#).

Ces fonctions permettent d'accéder à la librairie mcrypt, qui dispose d'une grande variété d'algorithmes de cryptage, tels que DES, TripleDES, Blowfish (par défaut), 3-WAY, SAFER-SK64, SAFER-SK128,

TWOFISH, TEA, RC2 et GOST en modes CBC, OFB, CFB et ECB. De plus, elle accepte aussi RC6 et IDEA qui sont considérés comme "non libre".

Si vous compilez PHP avec la librairie libmccrypt 2.4.x, les algorithmes suivants sont supportés : CAST, LOKI97, RIJNDAEL, SAFERPLUS, SERPENT ainsi que les chiffrements suivants : ENIGMA (cryptage), PANAMA, RC4 et WAKE. Avec libmccrypt 2.4.x un autre mode de chiffrement est disponible : nOFB.

Pour l'utiliser, téléchargez la librairie libmccrypt-x.x.tar.gz grâce par [ici](#) et suivez les instructions d'installations incluses. Vous aurez aussi besoin de compiler PHP avec le paramètre `--with-mcrypt` pour activer cette extension.

Mcrypt permet de crypter et de décrypter, en utilisant les méthodes mentionnées ci-dessus. Les 4 commandes importantes [mcrypt_cfb\(\)](#), [mcrypt_cbc\(\)](#), [mcrypt_ecb\(\)](#) et [mcrypt_ofb\(\)](#) peuvent toutes opérer en mode MCRYPT_ENCRYPT et MCRYPT_DECRYPT.

Crypte une valeur avec un TripleDES, en mode ECB.

```
<?php
$key = "Cette cle est ultra secrete";
$input = "Rencontrons nous dans notre place secrete a 9 h 00.";
$encrypted_data = mcrypt_ecb(MCRYPT_TripleDES, $key, $input, MCRYPT_ENCRYPT);
?>
```

Cet exemple va retourner les données cryptées dans la variable \$encrypted_data.

Si vous avez compilé PHP avec libmccrypt 2.4.x, ces fonctions sont toujours disponibles, mais il est vivement conseillé d'utiliser les nouvelles fonctions avancées.

Encryption d'une valeur avec TripleDES sous 2.4.x en mode ECB

```
<?php
$key = "Ceci est une vraie cle secrete";
$input = "Rendez vous à 9 heures, dans notre planque.";
$td = mcrypt_module_open (MCRYPT_TripleDES, "", MCRYPT_MODE_ECB, "");
$iv = mcrypt_create_iv (mcrypt_enc_get_iv_size ($td), MCRYPT_RAND);
mcrypt_generic_init ($td, $key, $iv);
$encrypted_data = mcrypt_generic ($td, $input);
mcrypt_generic_end ($td);
?>
```

Cet exemple va retourner les données cryptées dans la variable \$encrypted_data.

Mcrypt peut opérer en 4 modes de cryptage (CBC, OFB, CFB, et ECB). Nous allons présenter la technique d'utilisation de ces modes. Pour plus de références et de détails, reportez vous au livre suivant : Applied Cryptography par Schneier (ISBN 0-471-11709-9).

- ECB (electronic codebook) ECB (electronic codebook) est prévu pour des données aléatoires, telles que des clés. Etant donné que les données sont peu nombreuses et aléatoires, les inconvénients de l'ECB ont ici un effet négatif favorable.
- CBC (cipher block chaining) est spécialement pratique avec les fichiers dont la sécurité ECB n'est pas suffisante.
- CFB (cipher feedback) est la meilleure méthode pour crypter des flots d'octets, quand les octets doivent être encryptés un par un.
- OFB (output feedback) est comparable à CFB, mais peut être utilisé lorsque des erreurs ne doivent pas être propagées.
- nOFB (output feedback, in nbit) est comparable à OFB, mais plus sûr, car il opère avec la taille de bloc de l'algorithme.
- STREAM est un mode supplémentaire, pour permettre l'utilisation d'algorithmes tels que WAKE ou RC4.

PHP ne supporte par encore le cryptage des flots d'octets. Pour l'instant, PHP n'accepte que le cryptage de chaîne.

Pour obtenir la liste complète des modes de chiffrement, reportez vous aux derniers `#define`, dans le fichier ``mccrypt.h'`. En règle générale, vous pouvez accéder à une méthode de chiffrement avec l'option `MCRYPT_nomDuChiffrement`.

Voici une liste non exhaustive des modes de chiffrements de l'extension `mccrypt`. Si un chiffrement n'est pas dans cette liste, mais disponible dans la librairie, vous pouvez supposer que cette documentation est hors d'âge.

- `MCRYPT_3DES`
- `MCRYPT_ARCFOUR_IV` (libmccrypt 2.4.x seulement)
- `MCRYPT_ARCFOUR` (libmccrypt 2.4.x seulement)
- `MCRYPT_BLOWFISH`
- `MCRYPT_CAST_128`
- `MCRYPT_CAST_256`
- `MCRYPT_CRYPT`
- `MCRYPT_DES`
- `MCRYPT_DES_COMPAT` (libmccrypt 2.2.x seulement)
- `MCRYPT_ENIGMA` (libmccrypt 2.4.x seulement, alias de `MCRYPT_CRYPT`)
- `MCRYPT_GOST`
- `MCRYPT_IDEA` (payant)
- `MCRYPT_LOKI97` (libmccrypt 2.4.x seulement)
- `MCRYPT_MARS` (libmccrypt 2.4.x seulement, payant)
- `MCRYPT_PANAMA` (libmccrypt 2.4.x seulement)
- `MCRYPT_RIJNDAEL_128` (libmccrypt 2.4.x seulement)
- `MCRYPT_RIJNDAEL_192` (libmccrypt 2.4.x seulement)
- `MCRYPT_RIJNDAEL_256` (libmccrypt 2.4.x seulement)
- `MCRYPT_RC2`
- `MCRYPT_RC4` (libmccrypt 2.2.x seulement)
- `MCRYPT_RC6` (libmccrypt 2.4.x seulement)
- `MCRYPT_RC6_128` (libmccrypt 2.2.x seulement)
- `MCRYPT_RC6_192` (libmccrypt 2.2.x seulement)
- `MCRYPT_RC6_256` (libmccrypt 2.2.x seulement)
- `MCRYPT_SAFER64`
- `MCRYPT_SAFER128`
- `MCRYPT_SAFERPLUS` (libmccrypt 2.4.x seulement)
- `MCRYPT_SERPENT` (libmccrypt 2.4.x seulement)
- `MCRYPT_SERPENT_128` (libmccrypt 2.2.x seulement)
- `MCRYPT_SERPENT_192` (libmccrypt 2.2.x seulement)
- `MCRYPT_SERPENT_256` (libmccrypt 2.2.x seulement)
- `MCRYPT_SKIPJACK` (libmccrypt 2.4.x seulement)
- `MCRYPT_TEAN` (libmccrypt 2.2.x seulement)
- `MCRYPT_THREEWAY`
- `MCRYPT_TRIPLEDES` (libmccrypt 2.4.x seulement)
- `MCRYPT_TWOFISH` (Pour les anciennes versions de `mccrypt` 2.x versions, ou `mccrypt` 2.4.x)
- `MCRYPT_TWOFISH128` (`TWOFISHxxx` sont disponibles avec les nouvelles versions de 2.x, mais pas dans les versions 2.4.x)
- `MCRYPT_TWOFISH192`
- `MCRYPT_TWOFISH256`
- `MCRYPT_WAKE` (libmccrypt 2.4.x seulement)

- MCRYPT_XTEA (libmcrypt 2.4.x seulement)

Vous devez (mode CFB et OFB) ou pouvez (mode CBC) fournir un vecteur d'initialisation (IV) pour ces modes de chiffrement. IV doit être unique, et avoir la même valeur au chiffrement et au déchiffrement. Pour des données qui seront enregistrées après encryptage, vous pouvez prendre le résultat d'une fonction telle que MD5, appliquée sur le nom du fichier. Sinon, vous pouvez envoyer IV avec les données chiffrées, (reportez vous au chapitre 9.3 de Applied Cryptography by Schneier (ISBN 0-471-11709-9) pour plus de détails sur le sujet).

10.40.1 mcrypt_get_cipher_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_get_cipher_name](#)(int *cipher*)
[PHP 3>= 3.0.8, PHP 4]

string [mcrypt_get_cipher_name](#)(string *cipher*)
[PHP 3>= 3.0.8, PHP 4]

[mcrypt_get_cipher_name\(\)](#) retourne le nom du chiffrement utilisé.

[mcrypt_get_cipher_name\(\)](#) prend le numéro de chiffrement (avec libmcrypt 2.2.x) ou prend la nom du chiffrement (avec libmcrypt 2.4.x) comme paramètre, et retourne le nom du chiffrement, ou FALSE, si ce chiffrement n'existe pas.

Exemple avec mcrypt_get_cipher_name

```
<?php
$cipher = MCRYPT_TripleDES;
print mcrypt_get_cipher_name($cipher);
?>
```

L'exemple ci dessus va donner : TripleDES

10.40.2 mcrypt_get_block_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_get_block_size](#)(int *cipher*) int [mcrypt_get_block_size](#)|string *cipher*,
string *module*|
[PHP 3>= 3.0.8, PHP 4]

[mcrypt_get_block_size\(\)](#) sert à lire la taille de bloc du chiffrement *cipher*.

[mcrypt_get_block_size\(\)](#) prend comme argument le chiffrement *cipher* et retourne une taille en octets.
Voir aussi : [mcrypt_get_key_size\(\)](#).

10.40.3 mcrypt_get_key_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_get_key_size](#)(int *cipher*) int [mcrypt_get_key_size](#)|string *cipher*, string

module|

[PHP 3>= 3.0.8, PHP 4]

[mcrypt_get_key_size\(\)](#) sert à lire la taille de clé du chiffrement *cipher*.[mcrypt_get_block_size\(\)](#) prend comme argument le chiffrement *cipher* et retourne une taille en octets.Voir aussi: [mcrypt_get_block_size\(\)](#).**10.40.4 mcrypt_create_iv**[\[Notes en ligne\]](#) [\[Exemples\]](#)string [mcrypt_create_iv](#) (int *size*, int *source*)

[PHP 3>= 3.0.8, PHP 4]

[mcrypt_create_iv\(\)](#) sert à créer un IV (vecteur d'initialisation).[mcrypt_create_iv\(\)](#) prend deux arguments, *size* détermine la taille de IV, *source* spécifie la source de IV.

La source peut être MCRYPT_RAND (générateur de nombres aléatoires système),

MCRYPT_DEV_RANDOM (lecture des données depuis le fichier /dev/random) et

MCRYPT_DEV_URANDOM (lecture des données depuis le fichier /dev/urandom). Si vous utilisez

MCRYPT_RAND, assurez vous de bien appeler [srand\(\)](#) pour initialiser le générateur de nombres aléatoires.***Exemple avec mcrypt_create_iv***

```
<?php
$cipher = MCRYPT_TripleDES;
$block_size = mcrypt_get_block_size($cipher);
$iv = mcrypt_create_iv($block_size, MCRYPT_DEV_RANDOM);
?>
```

10.40.5 mcrypt_cbc[\[Notes en ligne\]](#) [\[Exemples\]](#)string [mcrypt_cbc](#) (int *cipher*, string *key*, string *data*, int *mode*, string *iv*)

[PHP 3>= 3.0.8, PHP 4]

string [mcrypt_cbc](#) (string *cipher*, string *key*, string *data*, int *mode*, string *iv*)

[PHP 3>= 3.0.8, PHP 4]

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_cbc\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode CBC et retourne la chaîne résultant.*Cipher* est une des constantes MCRYPT_ciphername*Key* est la clé fournie à l'algorithme. Elle doit être tenue secrète.*Data* sont les données à traiter.*Mode* vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.*IV* est le vecteur d'initialisation (optionnel).Voir aussi: [mcrypt_cfb\(\)](#), [mcrypt_ecb\(\)](#), et [mcrypt_ofb\(\)](#).

10.40.6 mcrypt_cfb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_cfb](#)(int *cipher*, string *key*, string *data*, int *mode*, string *iv*)
[PHP 3>= 3.0.8, PHP 4]

string [mcrypt_cfb](#)(string *cipher*, string *key*, string *data*, int *mode*, string *iv*)
[PHP 3>= 3.0.8, PHP 4]

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_cfb\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode CFB et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphernam

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mcrypt_cbc\(\)](#), [mcrypt_ecb\(\)](#), et [mcrypt_ofb\(\)](#).

10.40.7 mcrypt_ecb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_ecb](#)(int *cipher*, string *key*, string *data*, int *mode*)
[PHP 3>= 3.0.8, PHP 4]

string [mcrypt_ecb](#)(string *cipher*, string *key*, string *data*, int *mode*, string *iv*)
[PHP 3>= 3.0.8, PHP 4]

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_ecb\(\)](#) encrypte ou décrypte (suivant le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode CFB et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphernam

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mcrypt_cbc\(\)](#), [mcrypt_cfb\(\)](#), et [mcrypt_ofb\(\)](#).

10.40.8 mcrypt_ofb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_ofb](#)(int *cipher*, string *key*, string *data*, int *mode*, string *iv*)
[PHP 3>= 3.0.8, PHP 4]

string [mcrypt_ofb](#)(string *cipher*, string *key*, string *data*, int *mode*, string *iv*)
[PHP 3>= 3.0.8, PHP 4]

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_ofb\(\)](#) encode ou décode (selon le *mode* sélectionné) les données *data* avec le chiffrement *cipher* et la clé *key* en mode OFB et retourne la chaîne résultant.

Cipher est une des constantes MCRYPT_ciphername

Key est la clé fournie à l'algorithme. Elle doit être tenue secrète.

Data sont les données à traiter.

Mode vaut MCRYPT_ENCRYPT ou MCRYPT_DECRYPT.

IV est le vecteur d'initialisation (optionnel).

Voir aussi: [mcrypt_cbc\(\)](#), [mcrypt_cfb\(\)](#), et [mcrypt_ecb\(\)](#).

10.40.9 mcrypt_list_algorithms

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcrypt_list_algorithms](#)(string *lib_dir*)

[PHP 4 >= 4.0.2]

[mcrypt_list_algorithms\(\)](#) sert à lister tous les algorithmes de chiffrement de *lib_dir*.

[mcrypt_list_algorithms\(\)](#) prend un argument optionnel, qui spécifie le dossier qui contient tous les algorithmes. Si il est omis, la valeur de mcrypt.algorithms_dir dans `php.ini` est utilisée.

Exemple avec mcrypt_list_algorithms()

```
<?php
$algorithms = mcrypt_list_algorithms ("/usr/local/lib/libmcrypt");
foreach ($algorithms as $cipher) {
echo $cipher."/n";
}
?>
```

L'exemple ci dessus va affiche tous les algorithmes supportés dans le dossier "/usr/local/lib/libmcrypt".

10.40.10 mcrypt_list_modes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcrypt_list_modes](#)(string *lib_dir*)

[PHP 4 >= 4.0.2]

[mcrypt_list_algorithms\(\)](#) sert à lister tous les modes de chiffrement de *lib_dir*.

[mcrypt_list_algorithms\(\)](#) prend un argument optionnel, qui spécifie le dossier qui contient tous les algorithmes. Si il est omis, la valeur de mcrypt.algorithms_dir dans `php.ini` est utilisée.

Exemple avec mcrypt_list_modes()

```
<?php
$modes = mcrypt_list_modes ();
foreach ($modes as $mode) {
echo $mode."<br>";
}
?>
```


L'exemple ci dessus va affiche tous les modes supportés dans le dossier "/usr/local/lib/libmcrypt".

10.40.11 mcrypt_get_iv_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_get_iv_size](#) (string *cipher*, string *mode*) int [mcrypt_get_iv_size](#) |resource *td*
[PHP 4 >= 4.0.2]

La première syntaxe utilise libmcrypt 2.2.x, et la seconde utilise libmcrypt 2.4.x.

[mcrypt_get_iv_size\(\)](#) retourne la taille du Vecteur d'initialisation (VI). En cas d'erreur, la fonction retourne FALSE. Si le VI est ignoré dans le couple chiffrement/mode demandé, zéro est retourné.

Cipher est une constante MCRYPT_ciphernam qui indique le nom de l'algorithme sous forme de chaîne.

Mode est une constante MCRYPT_MODE_modename qui peut valoir : "ecb", "cbc", "cfb", "ofb", "nofb" ou "stream".

10.40.12 mcrypt_encrypt

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_encrypt](#) (string *cipher*, string *key*, string *data*, string *mode*, string *iv*)
[PHP 4 >= 4.0.2]

[mcrypt_encrypt\(\)](#) encrypte les données, et retourne les données cryptées.

Cipher est une constante MCRYPT_ciphernam qui indique le nom de l'algorithme sous forme de chaîne.

Key est la clé utilisée pour encrypter les données. Si elle est plus petite que nécessaire, elle sera complétée avec des '\0'.

Data sont les données qui doivent être encryptées. Si la taille des données n'est pas de la forme n * taille_de_bloc, elles seront complétées avec des '\0'. La valeur retournée peut être plus grande que la valeur d'origine.

Mode est une constante MCRYPT_MODE_modename qui peut valoir : "ecb", "cbc", "cfb", "ofb", "nofb" ou "stream".

IV (Vecteur d'initialisation) est utilisé pour les modes CBC, CFB, OFB, et dans certains algorithmes de mode STREAM. Si vous le fournissez par le VI, alors qu'il est nécessaire, la fonction affichera une alerte, et utilise un VI composé de caractères '\0'.

Exemple avec mcrypt_encrypt()

```
<?php
$iv = mcrypt_create_iv (mcrypt_get_iv_size (MCRYPT_RIJNDAEL_256, MCRYPT_MODE_ECB), MCRYPT_RAND);
$key = "Ceci est une clé secrète";
$text = "Rencontrons nous à 11 heures, derrière le monument";
echo strlen ($text)."\n";
$encrypttext = mcrypt_encrypt (MCRYPT_RIJNDAEL_256, $key, $text, MCRYPT_MODE_ECB, $iv);
echo strlen ($encrypttext)."\n";
?>
```

L'exemple ci dessus affichera : 42 64

10.40.13 [mcrypt_decrypt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_decrypt](#) (string *cipher*, string *key*, string *data*, string *mode*, string *iv*)
[PHP 4 >= 4.0.2]

Cipher est une constante MCRYPT_ciphername qui indique le nom de l'algorithme sous forme de chaîne. **Key** est la clé utilisée pour encrypter les données. Si elle est plus petite que nécessaire, elle sera complétée avec des '\0'.

Data sont les données qui doivent être encryptées. Si la taille des données n'est pas de la forme n * taille_de_bloc, elles seront complétées avec des '\0'. La valeur retournée peut être plus grande que la valeur d'origine.

Mode est une constante MCRYPT_MODE_modename qui peut valoir : "ecb", "cbc", "cfb", "ofb", "nofb" ou "stream".

IV (Vecteur d'initialisation) est utilisé pour les modes CBC, CFB, OFB, et dans certains algorithmes de mode STREAM. Si vous le fournissez par le VI, alors qu'il est nécessaire, la fonction affichera une alerte, et utilise un VI composé de caractères '\0'.

10.40.14 [mcrypt_module_open](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [mcrypt_module_open](#) (string *algorithm*, string *algorithm_directory*, string *mode*, string *mode_directory*)
[PHP 4 >= 4.0.2]

[mcrypt_module_open\(\)](#) ouvre le module de l'algorithme et du mode à utiliser. Le nom de l'algorithme est spécifié par le paramètre *algorithm* (par exemple : "twofish"), ou bien une des constantes MCRYPT_ciphername. La librairie est refermée en appelant @xref{ fonction.mcrypt-module-close , , mcrypt_module_close() }, mais il n'est pas nécessaire d'appeler cette fonction si [mcrypt_generic_end\(\)](#) est utilisé. Normalement, [mcrypt_module_open\(\)](#) retourne un pointeur d'encryption, ou bien FALSE en cas d'erreur.

algorithm_directory et *mode_directory* servent à repérer les modules d'encryption. Si vous fournissez un nom de dossier, il sera utilisé. Si vous passez une chaîne vide (""), la valeur utilisée par *mcrypt.algorithms_dir* ou *mcrypt.modes_dir* sera celle indiquée dans les directives de configuration. Lorsque ces paramètres ne sont pas fournis les valeurs par défaut, compilées avec la librairie sont utilisées. (généralement /usr/local/lib/libmcrypt).

Exemple avec mcrypt_module_open()

```
<?php
$td = mcrypt_module_open (MCRYPT_DES, "", MCRYPT_MODE_ECB, "/usr/lib/mcrypt-modes");
?>
```

L'exemple ci dessus va essayer d'ouvrir le module de chiffrement par DES, dans le dossier par défaut, et le mode EBC dans le dossier /usr/lib/mcrypt-modes.

10.40.15 mdecrypt_generic_init

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mdecrypt_generic_init](#)(resource *td*, string *key*, string *iv*)

[PHP 4 >= 4.0.2]

La taille maximale de la clé doit être celle retournée par [mdecrypt_enc_get_key_size\(\)](#) et toutes les valeurs inférieures seront aussi valides. Le vecteur d'initialisation (VI) doit avoir la taille d'un bloc, mais vous devez lire sa taille en appelant [mdecrypt_enc_get_iv_size\(\)](#). IV est ignoré en mode ECB. IV DOIT exister en modes CFB, CBC, STREAM, nOFB et OFB. Il doit être aléatoire et unique (mais pas secret). Le même VI doit être utilisé pour le cryptage et le décryptage. Si vous ne voulez pas l'utiliser, remplissez-le de zéros, mais ce n'est pas recommandé. La fonction retourne (-1) en cas d'erreur.

Vous devez appeler [mdecrypt_generic_init\(\)](#) avant chaque appel à [mdecrypt_generic\(\)](#) ou [mdecrypt_generic\(\)](#).

10.40.16 mdecrypt_generic

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mdecrypt_generic](#)(resource *td*, string *data*)

[PHP 4 >= 4.0.2]

[mdecrypt_generic\(\)](#) crypte des données. Les données sont complétées par des "\0" pour obtenir une taille de n fois la taille d'un bloc. Elle retourne les données encryptées. Notez que la longueur de la chaîne retournée peut être plus longue que celle passée en argument, à cause du complément.

10.40.17 mdecrypt_generic

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mdecrypt_generic](#)(resource *td*, string *data*)

[PHP 4 >= 4.0.2]

[mdecrypt_generic\(\)](#). Notez que la longueur de la chaîne décryptée peut être plus longue que la chaîne originale, car elle peut avoir été complétée par des "\0".

Exemple avec mdecrypt_generic()

```
<?php
$iv_size = mdecrypt_enc_get_iv_size ($td);
$iv = @mdecrypt_create_iv ($iv_size, MCRYPT_RAND);
if (@mdecrypt_generic_init ($td, $key, $iv) != -1)
{
    $c_t = mdecrypt_generic ($td, $plain_text);
    @mdecrypt_generic_init ($td, $key, $iv);
    $p_t = mdecrypt_generic ($td, $c_t);
}
if (strcmp ($p_t, $plain_text, strlen($plain_text)) == 0)
echo "ok";
else
echo "erreur";
?>
```

L'exemple ci dessus montre comment vérifier que les données avant cryptage sont bien les mêmes que celles

après cryptage/décryptage.

[10.40.18 mcrypt_generic_end](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_generic_end](#)(resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_generic_end\(\)](#) termine le cryptage désigné par le pointeur *td*. En fait, elle supprime tous les buffers, et ferme les modules utilisés. Elle retourne FALSE en cas d'erreur, et TRUE sinon.

[10.40.19 mcrypt_enc_self_test](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_self_test](#)(resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_self_test\(\)](#) effectue un test du module ouvert et désigné par *td*. Si le test est concluant, elle retourne 0, sinon, 1.

[10.40.20 mcrypt_enc_is_block_algorithm_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_is_block_algorithm_mode](#)(resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_is_block_algorithm_mode\(\)](#) retourne 1 si ce mode utilise des algorithmes par blocs, et 0 sinon. (i.e. 0 pour stream, et 1 pour cbc, cfb, ofb).

[10.40.21 mcrypt_enc_is_block_algorithm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_is_block_algorithm](#)(resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_is_block_algorithm\(\)](#) retourne 1 si l'algorithme utilisé est un algorithme par bloc, et 0 si c'est un algorithme par flot.

[10.40.22 mcrypt_enc_is_block_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_is_block_mode](#)(resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_is_block_mode\(\)](#) retourne 1 si le mode retourne des blocs d'octets, ou bien 0 si il retourne des octets (par flot). (i.e. 1 pour cbc et ecb, et 0 pour cfb et stream).

[10.40.23 mcrypt_enc_get_block_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_get_block_size](#) (resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_get_block_size\(\)](#) retourne la taille de blocs d'un algorithme en octets.

[10.40.24 mcrypt_enc_get_key_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_get_key_size](#) (resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_get_key_size\(\)](#) retourne la taille maximale de clé acceptée par le mode désigné par *td*, en octets.

[10.40.25 mcrypt_enc_get_supported_key_sizes](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mcrypt_enc_get_supported_key_sizes](#) (resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_get_supported_key_sizes\(\)](#) retourne un tableau contenant les tailles des clés supportées par l'algorithme désigné par *td*. Si il retourne un tableau vide, c'est que toutes les clés entre 1 et [mcrypt_enc_get_key_size\(\)](#) sont acceptées par l'algorithme.

[10.40.26 mcrypt_enc_get_iv_size](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_enc_get_iv_size](#) (resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_get_iv_size\(\)](#) retourne la taille du VI de l'algorithme désigné par *td*, en octets. Si la valeur retournée est 0, c'est que l'algorithme ne demande pas de VI. Un VI est demandé en mode cbc, cfb et ofb, et parfois en mode stream.

[10.40.27 mcrypt_enc_get_algorithms_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_enc_get_algorithms_name](#) (resource *td*)

[PHP 4 >= 4.0.2]

[mcrypt_enc_get_algorithms_name\(\)](#) retourne le nom de l'algorithme désigné par *td*.

[10.40.28 mcrypt_enc_get_modes_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mcrypt_enc_get_modes_name](#)(resource *td*)
[PHP 4 >= 4.0.2]

[mcrypt_enc_get_modes_name\(\)](#) retourne le nom du mode désigné par *td*.

[10.40.29 mcrypt_module_self_test](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_self_test](#)(string *algorithm*, string *lib_dir*)
[PHP 4 >= 4.0.2]

[mcrypt_module_self_test\(\)](#) effectue un test sur l'algorithme spécifié. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système. Retourne TRUE si le test fonctionne, et FALSE sinon.

[10.40.30 mcrypt_module_is_block_algorithm_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_is_block_algorithm_mode](#)(string *mode*, string *lib_dir*)
[PHP 4 >= 4.0.2]

[mcrypt_module_is_block_algorithm_mode\(\)](#) retourne TRUE si le mode doit être utilisé avec un algorithme par bloc, sinon retourne 0 (i.e. 0 pour stream, et 1 pour cbc, cfb, ofb). Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.40.31 mcrypt_module_is_block_algorithm](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_is_block_algorithm](#)(string *algorithm*, string *lib_dir*)
[PHP 4 >= 4.0.2]

[mcrypt_module_is_block_algorithm\(\)](#) retourne TRUE si *algorithm* est un algorithme par bloc, sinon retourne 0. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

[10.40.32 mcrypt_module_is_block_mode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [mcrypt_module_is_block_mode](#)(string *mode*, string *lib_dir*)
[PHP 4 >= 4.0.2]

[mcrypt_module_is_block_mode\(\)](#) retourne TRUE si ce mode fournit des blocs d'octets, ou bien un flot d'octets. (i.e. 1 pour cbc et ecb, et 0 pour cfb et stream). Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

10.40.33 mcrypt_module_get_algo_block_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_module_get_algo_block_size](#) (string *algorithm*, string *lib_dir*)

[PHP 4 >= 4.0.2]

[mcrypt_module_get_algo_block_size\(\)](#) retourne la taille de bloc d'un algorithme, en octets. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

10.40.34 mcrypt_module_get_algo_key_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mcrypt_module_get_algo_key_size](#) (string *algorithm*, string *lib_dir*)

[PHP 4 >= 4.0.2]

[mcrypt_module_get_algo_key_size\(\)](#) retourne la taille maximale de la clé supporté par l'algorithme *algorithm*. Le paramètre optionnel *lib_dir* contient le chemin jusqu'au module de l'algorithme sur le système.

10.40.35 mcrypt_module_get_algo_supported_key_sizes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array @xref{function.mcrypt-module-enc-get-algo-supported-key-sizes , ,
[mcrypt_module_enc_get_algo_supported_key_sizes](#) (string *algorithm*, string *lib_dir*) }

@xref{function.mcrypt-module-enc-get-algo-supported-key-sizes , ,
[mcrypt_module_enc_get_algo_supported_key_sizes\(\)](#)} retourne un tableau avec les tailles des clés supportées par l'algorithme *algorithm*. Si le tableau retourné est vide, c'est que toutes les tailles de clé entre 1 et [mcrypt_module_get_algo_key_size\(\)](#) sont supportées. Le paramètre optionnel *lib_dir* peut contenir le dossier du module sur le système.

10.41 Hash

[\[Notes en ligne\]](#)

Ces fonctions ont été prévues pour fonctionner avec [mhash](#).

Cet ensemble de fonctions représente une interface avec la librairie mhash. mhash accepte un grand nombre d'algorithmes différents, tels que MD5, SHA1, GOST, bien d'autres.

Pour l'utiliser, téléchargez les distributions de mhash depuis le site [web ici](#) et suivez les instructions d'installation incluses. Vous aurez besoin de recompiler PHP avec l'option `--with-mhash` pour activer cette extension.

mhash sert à calculer des sommes de vérifications, des signatures de messages, etc...

Calcule un hash de type SHA1 et l'affiche au format hexadécimal

```
<?php
$input = "Rencontrons nous à 9h00 dans notre repaire secret.";
$hash = mhash(MHASH_SHA1, $input);
print "Le hash est ".bin2hex($hash)."\n";
?>
```

Cela va produire quelque chose du type (Note du Traducteur : c'est le hash de la version anglaise) Le hash est d3b85d710d8f6e4e5efd4d5e67d041f9cecedafe Pour avoir une liste complète des hash supportés, reportez vous à la documentation de mhash. En règle générale, vous pouvez utiliser un algorithme de hash avec le type : MHASH_NOMDEHASH. Par exemple pour utiliser HAVAL vous devez spécifier la constante PHP MHASH_HAVAL.

Voici une liste de hash qui sont actuellement supportés par mhash. Si un hash n'est pas dans la liste, mais qu'il est disponible avec mhash, c'est que ce document a pris de l'âge.

- MHASH_MD5
- MHASH_SHA1
- MHASH_HAVAL
- MHASH_RIPEMD160
- MHASH_RIPEMD128
- MHASH_SNEFRU
- MHASH_TIGER
- MHASH_GOST
- MHASH_CRC32
- MHASH_CRC32B

10.41.1 mhash_get_hash_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mhash_get_hash_name](#)(int *hash*)

[PHP 3>= 3.0.9, PHP 4]

[mhash_get_hash_name\(\)](#) sert à connaître le nom d'un hash.

[mhash_get_hash_name\(\)](#) prend un numéro d'identifiant de hash, et retourne son nom, ou bien FALSE si le hash n'existe pas, ou si une erreur est survenue.

Exemple mhash_get_hash_name()

```
<?php
$hash = MHASH_MD5;
print mhash_get_hash_name($hash);
?>
```

L'exemple ci dessus va afficher : MD5

10.41.2 mhash_get_block_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mhash_get_block_size](#)(int *hash*)

[PHP 3>= 3.0.9, PHP 4]

[mhash_get_block_size\(\)](#) sert à connaître la taille de bloc du hash spécifié *hash*.

[mhash_get_block_size\(\)](#) prend un seul argument : le *hash* et retourne la taille en octets, ou bien FALSE si le *hash* n'existe pas.

10.41.3 mhash_count

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mhash_count](#)(void)

[PHP 3>= 3.0.9, PHP 4]

[mhash_count\(\)](#) retourne l'identifiant de hash maximal. Les hash sont numérotés de 0 jusqu'à cet identifiant.

Parcourir la liste des hash

```

<?php
$nr = mhash_count();
for($i = 0; $i <= $nr; $i++) {
echo sprintf("The blocksize of %s is %d\n",
mhash_get_hash_name($i),
mhash_get_block_size($i));
}
?>

```

10.41.4 mhash

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mhash](#)(int *hash*, string *data*)

[PHP 3>= 3.0.9, PHP 4]

[mhash\(\)](#) applique la fonction de hash *hash* aux données *data* et retourne le résultat.

10.41.5 mhash_keygen_s2k

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mhash_keygen_s2k](#)(int *hash*, string *password*, string *salt*, int *bytes*)

[mhash_keygen_s2k\(\)](#) génère une clé de *bytes* octets de long, à partir d'un mot de passe. Cette fonction utilise l'algorithme Salted S2K, spécifié dans OpenPGP (RFC 2440). Cet algorithme va utiliser l'algorithme de hachage *hash* pour créer la clé. Le paramètre *salt* doit être différent et suffisamment aléatoire pour chaque clé que vous générez, afin de créer des clés différentes. Ce grain de sel reservira lorsque vous vérifierez les clés : c'est alors une bonne idée que de l'ajouter à la fin de la clé générée. *salt* doit avoir la longueur de 8 octets, et sera complété par des 0 si vous ne fournissez pas suffisamment de données. N'oubliez pas que les mots de passe fournis par les utilisateurs ne sont pas conseillé pour faire des clés cryptographique, étant donné que les utilisateurs normaux retiennent des mots de passe qu'ils peuvent saisir au clavier. Ces mots de passe utilisent uniquement 6 à 7 des 8 bits d'un caractère (voir moins). Il est vivement recommandé d'appliquer une fonction de transformation (comme celle-ci), à un mot de passe utilisateur.

10.42 Fonctions diverses

[\[Notes en ligne\]](#)

Ces fonctions ont été placées là, car elles ne rentraient dans aucune catégorie adéquate.

10.42.1 connection_aborted

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connection_aborted](#)(void)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b4]

[connection_aborted\(\)](#) retourne TRUE si le client a abandonné la connexion. Reportez vous à [8.1 Gestion des connexions](#) du chapitre [8 Caractéristiques](#).

10.42.2 connection_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connection_status](#)(void)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b4]

[connection_status\(\)](#) retourne les bits de status de la connexion. Reportez vous à la section [8.1 Gestion des connexions](#) pour plus de détails.

10.42.3 connection_timeout

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connection_timeout](#)(void)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b4]

[connection_timeout\(\)](#) retourne TRUE si le script a expiré. Reportez vous à la section [8.1 Gestion des connexions](#) pour plus de détails.

10.42.4 define

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [define](#)(string *name*, mixed *value*, int *case_insensitive*)

[PHP 3, PHP 4]

[define\(\)](#) définit une constante, de la même façon qu'une variable, sauf que :

- Les constantes ne commencent pas par le signe '\$'
- Les constantes sont accessibles partout, de manière globale.
- Les constantes ne peuvent pas être redéfinies, ou indéfinies, une fois qu'elles ont été définies.
- Les constantes ne représentent que des valeurs scalaires : il n'est pas possible de définir des tableaux ou des objets.

Le nom de la constante est donnée par le paramètre *name*; sa valeur est donnée par *value*.

Le troisième paramètre optionnel *case_insensitive* peut prendre la valeur de 1. Dans ce cas, le nom de la constante sera insensible à la casse (c'est la valeur par défaut). Cela signifie que, par défaut, CONSTANT et Constant représentent des valeurs différentes.

Définition d'une constante

```
<?php
define ("CONSTANT", "Bonjour le monde.");
echo CONSTANT; // affiche "Bonjour le monde."
?>
```

[define\(\)](#) retourne TRUE en cas de succès et FALSE sinon.
Voir aussi [defined\(\)](#) et la section sur les [9.2 Les constantes](#).

[10.42.5 defined](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [defined](#)(string *name*)
[PHP 3, PHP 4]

[defined\(\)](#) retourne TRUE si la constante nommée *name* a été définie, et FALSE sinon.
Voir aussi [define\(\)](#) et la section sur les [9.2 Les constantes](#).

[10.42.6 die](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [die](#)(string *message*)

[die\(\)](#) affiche la chaîne passée en paramètre, puis termine l'exécution du script. Il ne retourne rien de plus.

Exemple avec die()

```
<?php
$filename = '/path/to/data-file';
$file = fopen ($filename, 'r')
or die("impossible d'ouvrir le fichier ($filename)");
?>
```

[10.42.7 eval](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [eval](#)(string *code_str*)

[eval\(\)](#) évalue la chaîne *code_str* comme un script PHP. Parmi les utilisations possibles, cette fonction permet de stocker du code dans une base de données, pour utilisation ultérieure.

Il faut bien garder en tête que le code passé à [eval\(\)](#) doit être valide, y compris les points virgules de fin de ligne et les séquences d'échappement, sinon l'exécution se terminera.

N'oubliez pas que les variables utilisées dans la fonction [eval\(\)](#) resteront accessibles dans le script principal.

Exemple avec eval() – inclusion de texte

```
<?php
$string = 'tasse';
```

```
$name = 'cafe';
$str = 'Ceci est une $string avec mon $name dedans.<br>';
echo $str;
eval( "\$str = \"\$str\";\" );
echo $str;
?>
```

L'exemple ci dessus devrait afficher : Ceci est une \$string avec mon \$name dedans.
Ceci est une tasse avec mon cafe dedans.

10.42.8 exit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [exit](#)

[exit\(\)](#) termine l'analyse d'un script en cours d'exécution. Elle ne renvoie aucune valeur.

10.42.9 get_browser

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [get_browser](#) (string *user_agent*)

[PHP 3, PHP 4]

[get_browser\(\)](#) essaie de determiner les capacités du navigateur client. Cela se fait en lisant les informations dans le fichier `browscap.ini'. Par défaut, la valeur de \$HTTP_USER_AGENT est utilisée. Cependant, vous pouvez passer n'importe quelle valeur avec le paramètre optionnel *user_agent* à [get_browser\(\)](#).

Les informations sont retournées sous forme d'un objet, dont les différents membres contiendront des informations, telles que les versions majeures et mineures et des chaînes d'identification; des booléens pour des caractéristiques telles que frames, JavaScript, et cookies; et ainsi de suite.

Même si `browscap.ini' contient des informations sur de nombreux clients, il compte sur les utilisateurs pour être mis à jour. Le format du fichier est facilement compréhensible.

L'exemple suivant montre comment on peut lister les informations disponibles :

Exemple avec get_browser()

```
<?php
function list_array ($array) {
while (list ($key, $value) = each ($array)) {
    $str .= "<B>$key:</B> $value<br>\n";
}
return $str;
}
echo "$HTTP_USER_AGENT<hr>\n";
$browser = get_browser();
echo list_array ((array) $browser);
?>
```

L'affichage devrait ressembler à ceci :

```
Mozilla/4.5 [en] (X11; U; Linux 2.2.9 i586)<hr>
<B>browser_name_pattern:</B> Mozilla/4\5.*<br> <B>parent:</B> Netscape
4.0<br> <B>platform:</B> Unknown<br> <B>majorver:</B> 4<br>
```

```
<B>minorver:</B> 5<br> <B>browser:</B> Netscape<br> <B>version:</B>
4<br> <B>frames:</B> 1<br> <B>tables:</B> 1<br> <B>cookies:</B> 1<br>
<B>backgroundsounds:</B> <br> <B>vbscript:</B> <br> <B>javascript:</B>
1<br> <B>javaapplets:</B> 1<br> <B>activexcontrols:</B> <br>
<B>beta:</B> <br> <B>crawler:</B> <br> <B>authenticcodeupdate:</B> <br>
<B>msn:</B> <br> Pour fonctionner, votre configuration 7.1.14 Directives de configuration du navigateur doit mener au fichier `browscap.ini`.
```

Pour plus d'informations, (y compris pour les endroits où charger le fichier `browscap.ini`), suivez la FAQ PHP à <http://www.php.net/FAQ.html>.

Note : *Browscap a été ajouté dans PHP version 3.0b2.*

10.42.10 highlight_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [highlight_file](#)(string *filename*)

[PHP 4]

[highlight_file\(\)](#) affiche la syntaxe colorisée du fichier *filename*, en utilisant les couleurs définies dans le moteur interne de PHP.

Colorisation d'URL

Pour configurer une URL qui peut coloriser n'importe quel script que vous lui passez, nous avons besoin d'utiliser la directive Apache "ForceType", pour générer une URL exploitable, puis utiliser la fonction [highlight_file\(\)](#) pour afficher un code propre.

Dans votre configuration HTTP `httpd.conf`, vous pouvez ajouter le code suivant :

```
<Location /source>
ForceType application/x-httpd-php
</Location>
```

Puis, faire un fichier appelé "source", que vous placez dans votre racine de site web.

```
<HTML>
<HEAD>
<TITLE>Affichage de Source</TITLE>
</HEAD>
<BODY BGCOLOR="white">
<?php
    $script = getenv ("PATH_TRANSLATED");
    if (!$script) {
        echo "<BR><B>ERROR: Script Name needed</B><BR>";
    } else {
        if (ereg("(\\.php|\\.inc)$", $script)) {
            echo "<H1>Source of: $PATH_INFO</H1><n<HR><n";
            highlight_file($script);
        }
    }
}
```

```

    } else {
echo "<H1>ERREUR: Seuls les noms de fichier PHP ou de fichiers PH inclus sont autorisés</H1>";
    }
}
echo "<HR>Traité: ".date("Y/M/d H:i:s",time());
?>
</BODY>
</HTML>

```

Alors, vous pourrez utiliser une URL telle que celle ci dessous pour afficher une version colorisée de votre script "/path/to/script.php".

`http://your.server.com/source/path/to/script.php`

Voir aussi [highlight_string\(\)](#), [show_source\(\)](#).

10.42.11 highlight_string

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [highlight_string](#)(string *str*)
[PHP 4]

[highlight_string\(\)](#) affiche la version colorisée de la chaîne *str*, en utilisant les couleurs définies dans le moteur interne de PHP.

Voir aussi [highlight_file\(\)](#), [show_source\(\)](#).

10.42.12 ignore_user_abort

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ignore_user_abort](#)(int *setting*)
[PHP 3>= 3.0.7, PHP 4 >= 4.0b4]

[ignore_user_abort\(\)](#) active l'option décidant si, lors de la déconnexion du client, le script doit poursuivre son exécution ou non. La fonction renvoie le paramétrage précédent et elle peut être appelée sans argument pour ne pas changer le paramétrage courant. Voir le paragraphe gestion des connexions dans le chapitre caractéristiques pour une description plus complète des manipulations de connexion en PHP.

10.42.13 iptcparse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [iptcparse](#)(string *iptcblock*)
[PHP 3>= 3.0.6, PHP 4]

[iptcparse\(\)](#) analyse un bloc binaire IPTC et recherche les balises simples. [iptcparse\(\)](#) retourne un tableau avec les balises comme index et les valeurs dans les valeurs de tableau correspondantes. En cas d'erreur, ou si aucune balise IPTC n'a été trouvée, retourne FALSE. Voir [getimagesize\(\)](#) pour un exemple.

[10.42.14 leak](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [leak](#)(int *bytes*)

[PHP 3, PHP 4]

[leak\(\)](#) crée une fuite de mémoire.

[leak\(\)](#) est pratique pour déboguer le gestionnaire de mémoire, qui doit nettoyer automatiquement les fuites de mémoire après chaque requête.

[10.42.15 pack](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pack](#)(string *format*, mixed *args* ...)

[PHP 3, PHP 4]

[pack\(\)](#) compacte les arguments dans une chaîne binaire, suivant le format *format*. [pack\(\)](#) retourne la chaîne binaire.

L'idée vient du Perl et tout le formatage fonctionne de la même fonction qu'en Perl, mais quelques formats manquent encore (comme, "u"). La chaîne de format est composée d'une série de code de formats, suivis par un quantificateur optionnel. Le quantificateur peut être un entier, ou * pour la répétition indéfinie. Pour les formats a, A, h et H, le quantificateur spécifie combien de caractères d'un argument sont pris; pour @, c'est la position absolue où placer les données, et pour le reste, c'est le nombre de répétitions. Actuellement, les formats suivants sont implémentés :

- Une chaîne complétée avec NULL
- Une chaîne complétée avec espace (SPACE)
- Chaîne hexadécimale h, bit de poids faible en premier.
- Chaîne hexadécimale H, bit de poids fort en premier.
- c caractère signé
- C caractère non signé
- s entier court signé (toujours sur 16 bits, ordre des bits dépendant de la machine).
- S entier court non signé (toujours 16 bits, ordre des bits dépendant de la machine).
- n entier court signé (toujours 16 bits, ordre des bits big endian)
- v entier cours non signé (toujours 16 bits, ordre des bits little endian)
- i entier signé (taille et ordre des bits dépendants de la machine)
- I entier non signé (taille et ordre des bits dépendants de la machine)
- l entier long signé (toujours 32 bits, ordre des bits dépendant de la machine)
- L entier long non signé (toujours 32 bits, ordre des bits dépendant de la machine)
- N entier long non signé (toujours 16 bits, ordre des bits big endian)
- V entier long non signé (toujours 16 bits, ordre des bits little endian)
- f nombre à virgule flottante (taille et représentation dépendantes de la machine)
- d nombre à virgule flottante double (taille et représentation dépendantes de la machine)
- x bit NULL
- X recule d'un octet
- rempli avec NULL, jusqu'à une position absolue

Compactage d'une chaîne

```
<?php
$binarydata = pack ("nvc*", 0x1234, 0x5678, 65, 66);
?>
```

La chaîne binaire résultante aura 6 octets de long, et contiendra la séquence 0x12, 0x34, 0x78, 0x56, 0x41, 0x42.

Notez que la distinction entre signé et non signé n'affecte que la fonction [unpack\(\)](#), tandis que la fonction [pack\(\)](#) fournira le même résultat pour les deux formats.

De plus, notez que PHP enregistre de manière interne et intégrale les valeurs : cette représentation dépend de la machine. Si vous essayez d'enregistrer une valeur trop grande, elle risque d'être convertie et de donner lieu à des effets de bords vicieux.

[10.42.16 show_source](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [show_source](#) (string *filename*)
[PHP 4]

[show_source\(\)](#) affiche la syntaxe colorisée du fichier *filename*, en utilisant les couleurs définies dans le moteur interne de PHP.

Note : [show_source\(\)](#) est un alias de [highlight_file\(\)](#).

Voir aussi [highlight_string\(\)](#), [highlight_file\(\)](#).

[10.42.17 sleep](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sleep](#) (int *seconds*)
[PHP 3, PHP 4]

[sleep\(\)](#) retarde l'exécution du programme pendant *seconds* secondes.

Voir aussi [usleep\(\)](#).

[10.42.18 uniqid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [uniqid](#) (string *prefix*, boolean *lcg*)
[PHP 3, PHP 4]

[uniqid\(\)](#) retourne un identifiant préfixé unique, basé sur l'heure courante, en micro-secondes. Le préfixe peut servir à identifier facilement différents hôtes, si vous générez simultanément des fichiers depuis plusieurs hôtes, à la même micro-seconde. *prefix* peut prendre jusqu'à 114 caractères.

Si le paramètre optionnel *lcg* est TRUE, [uniqid\(\)](#) ajoutera une entropie "combined LCG" à la fin de la valeur retournée, ce qui renforcera encore l'unicité de l'identifiant.

Sans *prefix* (préfixe vide), la chaîne retournée fera 13 caractères. Si *lcg* est à TRUE, elle fera 23 caractères.

Note : Le paramètre *lcg* est utilisé à partir de PHP 4 et PHP 3.0.13 et ultérieurs.

Si vous voulez utiliser un identifiant unique, ou bien gérer des cookies, il est recommandé d'utiliser un code tel que celui ci :


```
<?php
$token = md5 (uniqid ("")); // pas de section aléatoire.
$better_token = md5 (uniqid (rand())); // mieux, difficile à deviner
?>
```

Ceci va créer un identifiant de 32 caractère (un nombre hexadécimal de 128) qui sera très difficile à prédire.

[10.42.19 unpack](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [unpack](#) (string *format*, string *data*)

[PHP 3, PHP 4]

[unpack\(\)](#) déconditionne des données depuis une chaîne binaire avec le format *format*. Retourne un tableau contenant les éléments déconditionnés.

[unpack\(\)](#) se comporte légèrement différemment de la version Perl car les données déconditionnées sont stockées dans un tableau. Pour cela, il faut donner un nom à chaque format utilisé et les séparer par des slash (/).

Exemple avec unpack()

```
<?php
$array = unpack ("c2chars/nint", $binarydata);
?>
```

Le tableau résultant contiendra les entrées suivantes : "chars1", "chars2" et "int".

Pour plus de détails, reportez vous à : [pack\(\)](#)

Il faut noter que PHP gère les valeurs en interne sous forme signée. Si vous déconditionnez une valeur qui est aussi grande que la taille utilisée en interne par PHP, le résultat se trouvera être un nombre négatif, même si il a été déconditionné avec l'option " non signé ".

[10.42.20 usleep](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [usleep](#) (int *micro_seconds*)

[PHP 3, PHP 4]

[sleep\(\)](#) retarde l'exécution du programme pendant *micro_seconds* micro-secondes.

Voir aussi [sleep\(\)](#).

Note : [sleep\(\)](#) est inopérante sous Windows

[10.43 mSQL](#)

[\[Notes en ligne\]](#)

Ces fonctions vous permettent d'accéder aux bases de données mSQL. Pour cela, vous devez compiler PHP avec le support msql, en utilisant l'option de configuration `--with-msql[=dir]`. Par défaut, le chemin est `'/usr/local/Hughes'`.

Plus d'informations sur mSQL à <http://www.hughes.com.au/>.

10.43.1 [mysql](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql](#)(string *database*, string *query*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne un identifiant positif de résultat de requête, ou FALSE en cas d'erreur.

[mysql\(\)](#) sélectionne la base de données *database*, et y exécute la requête *query*. Si l'identifiant de connexion *link_identifier* n'est pas fourni, la fonction va rechercher un lien ouvert à un serveur mSQL, et sinon, il va tenter d'en créer une, avec [mysql_connect\(\)](#), sans argument.

10.43.2 [mysql_affected_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_affected_rows](#)(int *query_identifier*)

[PHP 3>= 3.0.6, PHP 4]

[mysql_affected_rows\(\)](#) retourne le nombre de lignes affectées par la dernière commande INSERT, UPDATE ou DELETE sur le serveur associé au *link_identifier*. Si ce dernier n'est pas précisé, la dernière connexion est utilisée.

Voir aussi: [mysql_query\(\)](#).

10.43.3 [mysql_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_close](#)(int *link_identifier*)

[PHP 3, PHP 4]

[mysql_close\(\)](#) retourne TRUE en cas de succès, FALSE en cas d'erreur.

[mysql_close\(\)](#) ferme la connexion au serveur de base de données mSQL référencé par l'identifiant fourni. Si aucun identifiant n'est fourni, la dernière connexion sera utilisée.

Notez bien qu'il n'est pas toujours nécessaire d'appeler cette fonction, car les connexions non persistantes seront automatiquement fermées à la fin du script.

[mysql_close\(\)](#) ne peut pas fermer les connexions persistantes, générées par [mysql_pconnect\(\)](#).

Voir aussi: [mysql_connect\(\)](#) et [mysql_pconnect\(\)](#).

10.43.4 [mysql_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_connect](#)(string *hostname* , string *hostname:port* , string *username* , string *password*)

[PHP 3, PHP 4]

[mysql_connect\(\)](#) retourne un identifiant de connexion positif en cas de succès, et FALSE sinon.

[mysql_connect\(\)](#) établit une connexion à un serveur mSQL. Le nom d'hôte est optionnel, et lorsqu'il manque, localhost est utilisé.

Si un deuxième appel est fait à [mysql_connect\(\)](#), avec les mêmes arguments, ce ne sera pas une nouvelle

connexion qui va être ouverte, mais l'ancienne connexion qui sera utilisée, et son identifiant sera retourné. Le lien au serveur sera fermé dès la fin du script, ou bien, manuellement, lors de l'appel de [mysql_close\(\)](#). Voir aussi [mysql_pconnect\(\)](#) et [mysql_close\(\)](#).

[10.43.5 mysql_create_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_create_db](#)(string *database name*, int *link_identifier*)
[PHP 3, PHP 4]

[mysql_create_db\(\)](#) essaie de créer une nouvelle base de données sur le serveur référencé par l'identifiant *link_identifier*.

Voir aussi: [mysql_drop_db\(\)](#).

[10.43.6 mysql_createdb](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_createdb](#)(string *database name*, int *link_identifier*)
[PHP 3, PHP 4]

Identique à [mysql_create_db\(\)](#).

[10.43.7 mysql_data_seek](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_data_seek](#)(int *query_identifier*, int *row_number*)
[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[mysql_data_seek\(\)](#) déplace le pointeur interne de résultat mSQL, et le place à l'offset donné. Le prochain appel à la fonction to [mysql_fetch_row\(\)](#) retournera cette ligne.

Voir aussi: [mysql_fetch_row\(\)](#).

[10.43.8 mysql_dbname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_dbname](#)(int *query_identifier*, int *i*)
[PHP 3, PHP 4]

[mysql_dbname\(\)](#) retourne le nom de la base de données enregistré en position *i* du pointeur de résultat retourné par la fonction [mysql_listdbs\(\)](#). La fonction [mysql_numrows\(\)](#) peut être utilisée pour déterminer le nombre de bases disponibles.

[10.43.9 mysql_drop_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_drop_db](#)(string *database_name*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[mysql_drop_db\(\)](#) essaie d'effacer une base de données entière sur le serveur référencé par l'identifiant fourni.

Voir aussi: [mysql_create_db\(\)](#).

10.43.10 mysql_dropdb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_drop_db\(\)](#).

10.43.11 mysql_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_error](#)()

[PHP 3, PHP 4]

Les erreurs générées par mSQL ne sont plus traitées comme des alertes. Au lieu de cela, elles sont stockées, et accessibles à partir de cette fonction.

10.43.12 mysql_fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fetch_array](#)(int *query_identifieur*, int *result_type*)

[PHP 3, PHP 4]

Retourne un tableau qui contient la ligne demandée, ou FALSE, si il n'y a pas d'autres lignes.

[mysql_fetch_array\(\)](#) est une version évoluée de [mysql_fetch_row\(\)](#). En plus d'enregistrer les données dans un tableau à indice numérique, il peut enregistrer les données dans un tableau associatif, en utilisant les noms des champs comme clés.

Le deuxième argument *result_type* de [mysql_fetch_array\(\)](#) est une constante, et peut prendre les valeurs suivantes : `MYSQL_ASSOC`, `MYSQL_NUM`, et `MYSQL_BOTH`.

Méfiez vous des requêtes qui retournent une ligne qui ne contient qu'un champs de valeur 0 (ou NULL, ou chaîne vide).

Il est important de noter que [mysql_fetch_array\(\)](#) est marginalement plus lent que [mysql_fetch_row\(\)](#), alors qu'elle apporte un confort d'utilisation appréciable.

Voir aussi [mysql_fetch_row\(\)](#).

10.43.13 mysql_fetch_field

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mysql_fetch_field](#)(int *query_identifieur*, int *field_offset*)

[PHP 3, PHP 4]

Retourne un objet contenant les informations sur un champs.

[mysql_fetch_field\(\)](#) sert à lire les informations sur les champs, dans certaines requêtes. Si l'offset du champs n'est pas spécifié, le prochain champs sera retourné.

Les propriétés de l'objet sont :

- name – nom de la colonne
- table – nom de la table à qui appartient la colonne.
- not_null – 1 si la colonne ne peut être NULL
- primary_key – 1 si la colonne est une clé primaire
- unique – 1 la colonne est une clé unique
- type – le type de la colonne

Voir aussi [mysql_field_seek\(\)](#).

10.43.14 [mysql_fetch_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fetch_object](#) (int *query_identifieur*, int *result_type*)

[PHP 3, PHP 4]

Retourne un objet, dont les propriétés seront affectées suivant les champs de la ligne lue, ou FALSE si il ne reste plus de lignes.

[mysql_fetch_object\(\)](#) est identique à [mysql_fetch_array\(\)](#), avec une différence : c'est un objet qui est retourné, à la place d'un tableau. Par conséquent, cela signifie que vous ne pouvez accéder aux valeurs que par les noms des champs, et non plus avec leur offset. (les nombres sont interdits dans les noms de propriétés)

L'argument optionnel *result_type* de [mysql_fetch_array\(\)](#) est une constante qui peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

[mysql_fetch_object\(\)](#) est aussi rapide que [mysql_fetch_array\(\)](#), et marginalement plus lente que [mysql_fetch_row\(\)](#) (la différence est non significative).

Voir aussi: [mysql_fetch_array\(\)](#) et [mysql_fetch_row\(\)](#).

10.43.15 [mysql_fetch_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_row](#) (int *query_identifieur*)

[PHP 3, PHP 4]

Retourne un tableau qui contient la ligne demandée, ou FALSE, si il n'y a plus de lignes à lire.

[mysql_fetch_row\(\)](#) retourne une ligne, extraite du résultat associé à l'identifiant de résultat *query_identifieur*.

La ligne est retournée sous la forme d'un tableau. Chaque résultat est enregistré dans un champs, indexé numériquement, à partir de 0.

Les appels ultérieurs à [mysql_fetch_row\(\)](#) retourneront les lignes suivantes, ou FALSE, lorsqu'il n'y aura plus de ligne.

Voir aussi: [mysql_fetch_array\(\)](#), [mysql_fetch_object\(\)](#), [mysql_data_seek\(\)](#) et [mysql_result\(\)](#).

10.43.16 [mysql_fieldname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_fieldname](#) (int *query_identifieur*, int *field*)

[PHP 3, PHP 4]

[mysql_fieldname\(\)](#) retourne le nom du champs à l'index *field*. *query_identifieur* est un identifiant de résultat, et

field est un index de champs. `mysql_fieldname($result, 2)`; retournera le nom du deuxième champs, dans le résultat associé à *query_identifieur*.

10.43.17 mysql_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_field_seek](#) (int *query_identifieur*, int *field_offset*)

[PHP 3, PHP 4]

Recherche l'offset du champs *field_offset*. Le prochain appel à [mysql_fetch_field\(\)](#) qui d'aura pas d'argument *field_offset*, retournera ce champs.

Voir aussi: [mysql_fetch_field\(\)](#).

10.43.18 mysql_fieldtable

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fieldtable](#) (int *query_identifieur*, int *field*)

[PHP 3, PHP 4]

Retourne le nom de la table d'où est le champs *field* a été extrait.

10.43.19 mysql_fieldtype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_fieldtype](#) (int *query_identifieur*, int *i*)

[PHP 3, PHP 4]

[mysql_fieldtype\(\)](#) est similaire à [mysql_fieldname\(\)](#). Les arguments sont identiques, mais c'est le type du champs qui est retourné. Cela produira un résultat tel que "int", "string" ou "real".

10.43.20 mysql_fieldflags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_fieldflags](#) (int *query_identifieur*, int *i*)

[PHP 3, PHP 4]

[mysql_fieldflags\(\)](#) retourne le flag du champs spécifié. Actuellement, il peut valoir soit "not null", "primary key", ou une combinaison des deux ou "" (chaîne vide).

10.43.21 mysql_fieldlen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_fieldlen](#) (int *query_identifieur*, int *i*)

[PHP 3, PHP 4]

[mysql_fieldlen\(\)](#) retourne la longueur d'un champs.

10.43.22 [mysql_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_free_result](#)(int *query_identifieur*)

[PHP 3, PHP 4]

[mysql_free_result\(\)](#) libère de la mémoire le résultat associé à l'identifiant de résultat *query_identifieur*. Lorsque PHP a terminé une requête, cette mémoire est libérée, ce qui fait que vous n'aurez pas besoin de cette fonction. Vous pouvez toujours l'utiliser pour vous assurer que vous n'utilisez pas trop de mémoire durant un script.

10.43.23 [mysql_freeresult](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_free_result\(\)](#).

10.43.24 [mysql_list_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_fields](#)(string *database*, string *tablename*)

[PHP 3, PHP 4]

[mysql_list_fields\(\)](#) lit les informations de la table *tablename*. Les arguments sont le nom de la base de données, *database* et le nom de la table *tablename*. Cette fonction retourne un identifiant de résultat qui sera utilisé avec [mysql_fieldflags\(\)](#), [mysql_fieldlen\(\)](#), [mysql_fieldname\(\)](#) et [mysql_fieldtype\(\)](#). Un identifiant de résultat est un entier positif. La fonction retourne -1 si une erreur survient. Une chaîne décrivant l'erreur sera placée dans la variable \$phperrormsg, et à moins que cette fonction n'ait été appelée avec (@mysql_list_fields()), alors cette erreur sera affichée.

Voir aussi [mysql_error\(\)](#).

10.43.25 [mysql_listfields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_list_fields\(\)](#).

10.43.26 [mysql_list_dbs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_dbs](#)

[PHP 3, PHP 4]

[mysql_list_dbs\(\)](#) retourne un pointeur de résultat, qui contiendra les noms des bases de données disponibles sur la connexion mSQL courante. Utilisez [mysql_dbname\(\)](#) pour passer en revue toutes les lignes.

[10.43.27 mysql_listdbs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_list_dbs\(\)](#).

[10.43.28 mysql_list_tables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_tables](#)(string *database*)

[PHP 3, PHP 4]

[mysql_list_tables\(\)](#) prend un nom de base de données, et fourni un résultat, un peu comme la fonction [mysql\(\)](#). La fonction [mysql_tablename\(\)](#) devrait être utilisée de préférence pour extraire les nom de table d'un pointeur de résultat.

[10.43.29 mysql_listtables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_list_tables\(\)](#).

[10.43.30 mysql_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_fields](#)(int *query_identifiant*)

[PHP 3, PHP 4]

[mysql_num_fields\(\)](#) retourne le nombre de champs du résultat associé à l'identifiant *query_identifiant*. Voir aussi: [mysql\(\)](#), [mysql_query\(\)](#), [mysql_fetch_field\(\)](#) et [mysql_num_rows\(\)](#).

[10.43.31 mysql_num_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_rows](#)(int *query_identifiant*)

[PHP 3, PHP 4]

[mysql_num_rows\(\)](#) retourne le nombre de lignes du résultat associé à l'identifiant *query_identifiant*. Voir aussi: [mysql\(\)](#), [mysql_query\(\)](#) et [mysql_fetch_row\(\)](#).

[10.43.32 mysql_numfields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_numfields](#)(int *query_identifiant*)

[PHP 3, PHP 4]

Identique à [mysql_num_fields\(\)](#).

[10.43.33 mysql_numrows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_numrows](#)

[PHP 3, PHP 4]

Identique à [mysql_num_rows\(\)](#).

[10.43.34 mysql_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_pconnect](#)(string *hostname* , string *hostname:port* , string *username* , string *password*)

[PHP 3, PHP 4]

Retourne un identifiant de connexion persistante à un serveur mSQL en cas de succès, et FALSE sinon.

[mysql_pconnect\(\)](#) se comporte presque comme [mysql_connect\(\)](#) mais avec deux différences majeures.

D'abord, lors de la connexion, [mysql_pconnect\(\)](#) cherche si une connexion persistante a déjà été ouverte sur le même hôte. Si une telle connexion est trouvée, elle sera utilisée.

Deuxièmement, la connexion au serveur SQL ne sera pas terminée lors de la fin de l'exécution du script. A la place, le lien restera ouvert pour d'autres connexions futures. ([mysql_close\(\)](#) ne fermera pas un lien ouvert par [mysql_pconnect\(\)](#)).

C'est pourquoi une telle connexion est considérée comme 'persistante'.

[10.43.35 mysql_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_query](#)(string *query*, int *link_identifiant*)

[PHP 3, PHP 4]

[mysql_query\(\)](#) envoie une requête à la base de donnée active, sur le serveur associé à l'identifiant de connexion *link_identifiant*. Si *link_identifiant* n'est pas fourni, PHP tentera d'utiliser la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction tentera de se connecter par elle même, avec

[mysql_connect\(\)](#) appelé sans argument.

Retourne un identifiant positif mSQL en cas de succès, et FALSE sinon.

Voir aussi: [mysql\(\)](#), [mysql_select_db\(\)](#), et [mysql_connect\(\)](#).

[10.43.36 mysql_regcase](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [sql_regcase\(\)](#).

[10.43.37 mysql_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_result](#)(int *query_identifiant*, int *i*, mixed *field*)

[PHP 3, PHP 4]

[mysql_result\(\)](#) retourne la valeur de la cellule, à la ligne *i* et l'offset spécifié, *field* dans le résultat mSQL *query_identifier*.

[mysql_result\(\)](#) retourne le contenu d'une cellule depuis un résultat mSQL *query_identifier*. L'argument de champs *field* peut être aussi bien un offset, qu'un nom de champs, ou encore le nom de la table point le nom du fichier (nom_table.nom_champs). Si la colonne est un alias, (par exemple 'select foo as bar from...'), utilisez de préférence l'alias au nom de colonne.

Lorsque vous travailler sur des résultats de grande taille, il est préférable d'utiliser les fonctions qui récupèrent toute la ligne (voir ci dessous). Comme ces fonctions retournent plusieurs cellules en même temps, elles sont beaucoup plus rapide que [mysql_result\(\)](#). De plus, sachez qu'accéder à un champs avec son indice numérique est beaucoup plus rapide qu'en utilisant les autres méthodes.

Alternatives recommandées : [mysql_fetch_row\(\)](#), [mysql_fetch_array\(\)](#) et [mysql_fetch_object\(\)](#).

10.43.38 mysql_select_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_select_db](#) (string *database_name*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[mysql_select_db\(\)](#) choisi la base de données courante sur le serveur associé à l'identifiant de connexion *link_identifier*. Si *link_identifier* n'est pas fourni, PHP tentera d'utiliser la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction tentera de se connecter par elle-même, avec [mysql_connect\(\)](#) appelée sans argument.

Les prochains appels à [mysql_query\(\)](#) seront fait dans la base de données active.

Voir aussi: [mysql_connect\(\)](#), [mysql_pconnect\(\)](#) et [mysql_query\(\)](#).

10.43.39 mysql_selectdb

[\[Notes en ligne\]](#) [\[Exemples\]](#)

Voir [mysql_select_db\(\)](#).

10.43.40 mysql_tablename

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_tablename](#) (int *query_identifier*, int *field*)

[PHP 3, PHP 4]

[mysql_tablename\(\)](#) prend un pointeur de résultat (retourné par la fonction [mysql_list_tables\(\)](#)), ainsi qu'un index, et retourne le nom d'une table. La fonction [mysql_numrows\(\)](#) peut servir à déterminer le nombre de table dans le pointeur de résultat.

Exemple [mysql_tablename\(\)](#)

```
<?php
mysql_connect ("localhost");
$result = mysql_list_tables ("limousin");
$i = 0;
while ($i < mysql_numrows ($result)) {
    $tb_names[$i] = mysql_tablename ($result, $i);
    echo $tb_names[$i] . "<BR>";
    $i++;
}
```

```
}
?>
```

10.44 Microsoft SQL Server

[\[Notes en ligne\]](#)

10.44.1 mssql_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_close](#) (int *link_identifiant*)
[PHP 3, PHP 4]

[mssql_close\(\)](#) retourne TRUE en cas de succès, ou FALSE sinon.

[mssql_close\(\)](#) ferme la connexion à la base MS SQL Server, qui était associé à l'identifiant *link_identifiant*. Si ce dernier n'est pas précisé, la dernière connexion ouverte sera fermée.

Notez qu'il n'est pas nécessaire de fermer les connexions non persistantes aux bases de données, car elles seront fermées automatiquement à la fin du script.

[mssql_close\(\)](#) ne peut pas fermer les liens persistants, générés par [mssql_pconnect\(\)](#).

Voir aussi : [mssql_connect\(\)](#) et [mssql_pconnect\(\)](#).

10.44.2 mssql_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_connect](#) (string *servername* , string *username* , string *password*)
[PHP 3, PHP 4]

[mssql_connect\(\)](#) retourne un identifiant positif de lien en cas de succès, et FALSE sinon.

[mssql_connect\(\)](#) établit une connexion à un serveur MS SQL. Le nom du serveur *servername* doit être valide, comme défini dans les fichiers d'interface.

Si un deuxième appel est fait à [mssql_connect\(\)](#) avec les mêmes arguments, un nouveau lien ne sera pas retourné, mais le lien déjà ouvert sera retourné.

Le lien avec le serveur sera fermé dès la fin du script, ce qui fait qu'on n'est pas obligé de fermer explicitement la connexion à la fin du script avec [mssql_close\(\)](#).

Voir aussi [mssql_pconnect\(\)](#), [mssql_close\(\)](#).

10.44.3 mssql_data_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_data_seek](#) (int *result_identifiant*, int *row_number*)
[PHP 3, PHP 4]

[mssql_data_seek\(\)](#) retourne TRUE en cas de succès, FALSE en cas d'échec.

[mssql_data_seek\(\)](#) déplace le pointeur interne de ligne, dans le résultat *result_identifiant*, jusqu'à la ligne *row_number*. Le prochain appel à [mssql_fetch_row\(\)](#) retournera cette ligne.

Voir aussi : [mssql_data_seek\(\)](#).

10.44.4 mssql_fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_fetch_array](#) (int *result*)

[PHP 3, PHP 4]

[mssql_fetch_array\(\)](#) retourne un tableau qui contient les valeurs de la ligne lues, en cas de succès, et FALSE en cas d'échec.

[mssql_fetch_array\(\)](#) est une version améliorée de [mssql_fetch_row\(\)](#). En plus de stocker les données dans un tableau à index numérique, elle les stocke aussi dans un tableau associatif, en utilisant les noms de colonnes comme clé.

Une chose importante à noter est que [mssql_fetch_array\(\)](#) n'est PAS significativement plus lente que [mssql_fetch_row\(\)](#), tandis qu'elle apporte un confort appréciable.

Pour plus de détails, voyez [mssql_fetch_row\(\)](#).

10.44.5 mssql_fetch_field

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mssql_fetch_field](#) (int *result*, int *field_offset*)

[PHP 3, PHP 4]

[mssql_fetch_field\(\)](#) retourne un objet contenant les informations sur un champs.

[mssql_fetch_field\(\)](#) sert à lire des informations spécifiques à un champs, dans un résultat de requête. Si l'offset du champs *field_offset* n'est pas précisé, le prochain champs sera analysé.

Les propriétés de l'objet sont :

- name – nom de la colonne. Si la colonne est le résultat d'une fonction, le nom de la colonne sera computed#N, où #N est un numéro de série.
- column_source – le nom de la table d'où la colonne est originaire.
- max_length – taille maximale de la colonne
- numeric – 1 si la colonne est numérique

Voir aussi [mssql_field_seek\(\)](#).

10.44.6 mssql_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_fetch_object](#) (int *result*)

[PHP 3, PHP 4]

[mssql_fetch_object\(\)](#) retourne un objet dont les propriétés contiennent les valeurs de la ligne, ou FALSE si il n'y a plus de ligne.

[mssql_fetch_object\(\)](#) est similaire à [mssql_fetch_array\(\)](#), avec une différence : un objet est retourné, au lieu d'un tableau. Indirectement, cela signifie que vous ne pouvez accéder aux données que par leur nom de champs, et pas par leur offset (les nombres sont illégaux comme nom de propriété).

En terme de vitesse, cette fonction est identique à [mssql_fetch_array\(\)](#), quasiment aussi rapide que [mssql_fetch_row\(\)](#) (la différence est non significative).

Voir aussi : [mssql_fetch_array\(\)](#) et [mssql_fetch_row\(\)](#).

10.44.7 mssql_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mssql_fetch_row](#) (int *result*)

[PHP 3, PHP 4]

[mssql_fetch_row\(\)](#) retourne un tableau qui contient les valeurs de la ligne à lire, ou bien FALSE si il n'y a plus de lignes à lire.

[mssql_fetch_row\(\)](#) lit une ligne dans le résultat *result* et place les valeurs dans un tableau. Chaque valeur est enregistré dans un élément du tableau, et les indices commencent à 0.

Les appels suivants à [mssql_fetch_row\(\)](#) retourneront la ligne suivante, ou bien FALSE s'il ne reste plus de lignes.

Voir aussi : [mssql_fetch_array\(\)](#), [mssql_fetch_object\(\)](#), [mssql_data_seek\(\)](#) et [mssql_result\(\)](#).

10.44.8 mssql_field_length

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_field_length](#) (int *result*, int *offset*)

[PHP 3>= 3.0.3, PHP 4 >= 4.0b4]

10.44.9 mssql_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_field_name](#) (int *result*, int *offset*)

[PHP 3>= 3.0.3, PHP 4 >= 4.0b4]

10.44.10 mssql_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_field_seek](#) (int *result*, int *field_offset*)

[PHP 3, PHP 4]

[mssql_field_seek\(\)](#) modifie la valeur du pointeur de champs. Le prochain appel à [mssql_fetch_field\(\)](#) qui ne précisera pas de numéro de champs, le champs fixé par [mssql_field_seek\(\)](#) sera retournée.

Voir aussi : [mssql_fetch_field\(\)](#).

10.44.11 mssql_field_type

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mssql_field_type](#) (int *result*, int *offset*)

[PHP 3>= 3.0.3, PHP 4 >= 4.0b4]

10.44.12 mssql_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_free_result](#) (int *result*)

[PHP 3, PHP 4]

[mysql_free_result\(\)](#) n'a besoin d'être appelé que si on craint d'utiliser trop de mémoire durant une opération. Toutes les ressources liées à un résultat seront libérés par [mysql_free_result\(\)](#).

10.44.13 mysql_get_last_message

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_get_last_message](#)(void)

[PHP 3, PHP 4]

10.44.14 mysql_min_error_severity

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mysql_min_error_severity](#)(int *severity*)

[PHP 3, PHP 4]

10.44.15 mysql_min_message_severity

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [mysql_min_message_severity](#)(int *severity*)

[PHP 3, PHP 4]

10.44.16 mysql_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_fields](#)(int *result*)

[PHP 3, PHP 4]

[mysql_num_fields\(\)](#) retourne le nombre de champs dans un résultat.

Voir aussi : [mysql_query\(\)](#), [mysql_fetch_field\(\)](#) et [mysql_num_rows\(\)](#).

10.44.17 mysql_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_rows](#)(string *result*)

[PHP 3, PHP 4]

[mysql_num_rows\(\)](#) retourne le nombre de lignes dans un résultat.

Voir aussi : [mysql_query\(\)](#) et [mysql_fetch_row\(\)](#).

10.44.18 mysql_pconnect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_pconnect](#)(string *servername* , string *username* , string *password*)

[PHP 3, PHP 4]

[mssql_pconnect\(\)](#) retourne un identifiant positif de lien MS SQL en cas de succès, et FALSE en cas d'erreur.

[mssql_pconnect\(\)](#) se comporte comme [mssql_connect\(\)](#) mais avec deux différences :

Premièrement, lors de la connexion, la fonction va commencer par rechercher un lien persistant déjà ouvert avec le même hôte, le même nom d'utilisateur, *username* et le même mot de passe *password*. Si un tel lien est trouvé, cet identifiant sera retourné, au lieu d'en ouvrir une autre connexion.

Deuxièmement, la connexion au serveur SQL ne sera pas fermée à la fin du script, mais restera ouverte, pour d'autres utilisations ultérieures ([mssql_close\(\)](#) ne fermera pas un lien établi avec [mssql_pconnect\(\)](#)).

C'est pourquoi ce type de lien est dit 'persistant'.

10.44.19 [mssql_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_query](#) (string *query*, int *link_identifieur*)

[PHP 3, PHP 4]

[mssql_query\(\)](#) retourne un identifiant positif de résultat en cas de succès, ou FALSE sinon.

[mssql_query\(\)](#) envoie la requête au serveur courant, associé à l'identifiant *link_identifieur* (ou la base par défaut, si il est omis). Si aucun lien n'est ouvert, [mssql_query\(\)](#) essaiera d'en ouvrir une, en appelant [mssql_connect\(\)](#).

Voir aussi : [mssql_select_db\(\)](#) et [mssql_connect\(\)](#).

10.44.20 [mssql_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_result](#) (int *result*, int *i*, mixed *field*)

[PHP 3, PHP 4]

[mssql_result\(\)](#) retourne la valeur de la colonne, à la ligne donnée, dans le résultat MS SQL, ou FALSE en cas d'erreur.

[mssql_result\(\)](#) retourne le contenu d'une des cellules d'un résultat MS SQL. Le nom du champs peut être son nom littéral ou son offset, ou encore, le nom de la table + "." + le nom du champs, ou encore la même chose avec le nom de la base de données. Si la colonne a été aliasée, utilisez le nom de l'alias plutôt que celui de la colonne.

Lorsque vous travaillez sur des résultats de grande taille, il vaut mieux utiliser les fonctions qui récupèrent toute une ligne (voir ci après). Comme ces fonctions lisent toutes les valeurs en une passe, elles sont EXTREMEMENT PLUS RAPIDES que [mssql_result\(\)](#). De plus, pensez que l'utilisation de l'offset numérique est beaucoup plus rapide que l'utilisation du nom de la colonne.

Alternatives recommandées : [mssql_fetch_row\(\)](#), [mssql_fetch_array\(\)](#) et [mssql_fetch_object\(\)](#).

10.44.21 [mssql_select_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mssql_select_db](#) (string *database_name*, int *link_identifieur*)

[PHP 3, PHP 4]

[mssql_select_db\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[mssql_select_db\(\)](#) sélectionne la base de données active. Si aucun identifiant de connexion n'est fourni, la fonction utilisera la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction essaiera d'en

ouvrir une avec [mysql_connect\(\)](#), et de l'utiliser.

Tous les appels à [mysql_query\(\)](#) seront faits dans cette base.

Voir aussi : [mysql_connect\(\)](#), [mysql_pconnect\(\)](#) et [mysql_query\(\)](#).

10.45 MySQL

[\[Notes en ligne\]](#)

Ces fonctions vous permettent d'accéder aux bases de données MySQL. Afin de pouvoir les utiliser, vous devez compiler PHP avec le support MySQL, en utilisant l'option `--with-mysql`. Si vous utilisez cette fonction sans préciser le chemin d'accès à la base MySQL, PHP utilisera les librairies cliente MySQL fournies en standard. Les utilisateurs qui font tourner d'autres applications qui utilisent elles-mêmes MySQL (par exemple, PHP 3 et PHP 4 utilisés comme des modules concurrents apache, ou encore auth-mysql), devrait toujours spécifier le chemin jusqu'à MySQL : `--with-mysql=/path/to/mysql`. Cela va forcer PHP à utiliser les librairies clientes installées par MySQL, et évitera les conflits.

Plus d'informations sont disponibles à <http://www.mysql.com/>.

La documentation de MySQL est disponibles à <http://www.mysql.com/documentation/>, ainsi qu'en français chez [nexen](#).

10.45.1 mysql_affected_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_affected_rows](#) (int *link_identifiant*)

[PHP 3, PHP 4]

[mysql_affected_rows\(\)](#) retourne le nombre de lignes affectées lors de la dernière requête INSERT, UPDATE ou DELETE sur le serveur associé à l'identifiant de connexion. Si cet identifiant n'est pas précisé, [mysql_affected_rows\(\)](#) utilise la dernière connexion ouverte.

Note : *Si vous utilisez les transactions, vous devez appeler [mysql_affected_rows\(\)](#) après votre INSERT, UPDATE, ou DELETE, et non après la validation.*

Si la dernière requête était un DELETE sans clause WHERE, tous les enregistrements ont été effacés, mais [mysql_affected_rows\(\)](#) va retourner 0.

[mysql_affected_rows\(\)](#) n'est pas possible après un SELECT, car elle ne fonctionne qu'après des commandes qui modifient les enregistrements. Pour connaître le nombre de lignes retournées par un SELECT, utilisez [mysql_num_rows\(\)](#).

Si la dernière requête a échoué, [mysql_affected_rows\(\)](#) retourne -1.

10.45.2 mysql_change_user

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_change_user](#) (string *user*, string *password*, string *database* , int *link_identifiant*)

[PHP 3>= 3.0.13]

[mysql_change_user\(\)](#) change l'utilisateur en cours de la session active, ou sur la connexion spécifiée avec l'option *link_identifiant*. Si une base est spécifiée, elle deviendra la base par défaut de l'utilisateur. Si une erreur de connexion survient, la connexion en cours restera active.

Note : *Cette fonction a été introduite dans PHP 3.0.13 et requiert MySQL 3.23.3 ou plus récent.*

10.45.3 [mysql_close](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_close](#) (int *link_identifieur*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE sinon.

[mysql_close\(\)](#) ferme la connexion au serveur MySQL associée à l'identifiant *link_identifieur* . Si cet identifiant n'est pas spécifié, cette commande s'applique à la dernière connexion ouverte.

Note : *Notez que cette commande n'est pas nécessaire, car toutes les connexions non persistantes seront automatiquement fermées à la fin du script.*

[mysql_close\(\)](#) ne ferme pas les connexions persistantes générées par [mysql_pconnect\(\)](#).

Exemple MySQL_close

```
<?php
    $link = mysql_connect ("kraemer", "marliesle", "secret") {
or die ("Impossible de se connecter");
    }
print ("Connexion réussie");
mysql_close ($link);
?>
```

Voir aussi [mysql_connect\(\)](#) et [mysql_pconnect\(\)](#).

10.45.4 [mysql_connect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_connect](#) (string *hostname :port :/path/to/socket* , string *username* , string *password*)

[PHP 3, PHP 4]

Retourne un identifiant positif de connexion en cas de succès, et sinon FALSE.

[mysql_connect\(\)](#) établit une connexion à un serveur MySQL. Tous les arguments sont optionnels, et s'ils manquent, les valeurs par défaut sont utilisées (('localhost', nom du propriétaire du process, mot de passe vide).

Le nom d'hôte peut aussi inclure un numéro de port, sous la forme : "hostname:port" ou un chemin jusqu'à une socket sous la forme ":/path/to/socket" pour l'hôte localhost. Note : *Le support des ":port" a été ajouté à partir de la version 3.0B4.*

Le support de ":/path/to/socket" a été ajouté à partir de la version 3.0.10.

Vous pouvez supprimer le message d'erreur de connexion en ajoutant un arobase " au nom de la fonction.

Si un second appel à [mysql_connect\(\)](#) est fait avec les mêmes arguments, PHP ne va pas ouvrir une nouvelle connexion, mais va retourner l'identifiant de la connexion déjà ouverte.

Le lien sera fermé automatiquement dès que l'exécution du script sera terminée, à moins d'être fermé explicitement avec [mysql_close\(\)](#).

Exemple MySQL connect

```
<?php
```

```

$link = mysql_connect ("kraemer", "marliesle", "secret") {
or die ("Connexion impossible");
}
print ("Connexion réussie");
mysql_close ($link);
?>

```

Voir aussi [mysql_pconnect\(\)](#) et [mysql_close\(\)](#).

10.45.5 mysql_create_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_create_db](#)(string *database name*, int *link_identifier*)

[PHP 3, PHP 4]

[mysql_create_db\(\)](#) tente de créer une nouvelle base de données sur le serveur associé à l'identifiant *link_identifier*, ou la dernière connexion ouverte.

Exemple de création de base MySQL

```

<?php
$link = mysql_pconnect ("kron", "jutta", "geheim") {
or die ("Connexion impossible");
}
if (mysql_create_db ("my_db")) {
print ("Base de données créée\n");
} else {
printf ("Erreur lors de la création de la base: %s\n", mysql_error());
}
?>

```

Pour des raisons de compatibilité ascendante, [mysql_createdb\(\)](#) est toujours utilisable.

Voir aussi [mysql_drop_db\(\)](#).

10.45.6 mysql_data_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_data_seek](#)(int *result_identifier*, int *row_number*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE sinon.

[mysql_data_seek\(\)](#) déplace le pointeur interne de résultat, dans le résultat associé à l'identifiant de résultat *result_identifier*. Il le fait pointer à la ligne *row_number*. Le prochain appel à [mysql_fetch_row\(\)](#) retournera cette ligne.

Row_number commence à 0.

Exemple mysql_data_seek()

```

<?php
$link = mysql_pconnect ("kron", "jutta", "geheim") {
or die ("Connexion impossible");
}
mysql_select_db ("samp_db") {

```

```

or die ("Selection de base impossible");
}
$query = "SELECT last_name, first_name FROM friends";
$result = mysql_query ($query) {
or die ("Requête impossible");
}
# récupère les lignes dans l'ordre inverse
for ($i = mysql_num_rows ($result) - 1; $i >=0; $i--) {
if (!mysql_data_seek ($result, $i)) {
printf ("Impossible d'atteindre la ligne %d\n", $i);
continue;
}
if(!($row = mysql_fetch_object ($result)))
continue;
printf ("%s %s<BR>\n", $row->last_name, $row->first_name);
}
mysql_free_result ($result);
?>

```

10.45.7 mysql_db_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_db_name](#)(int *result*, int *row*, mixed *field*)

[PHP 3>= 3.0.6, PHP 4]

[mysql_db_name\(\)](#) prend comme premier argument le pointeur de résultat issu de [mysql_list_dbs\(\)](#), *row* est l'index dans le résultat.

Si une erreur survient, FALSE est retourné. Utilisez [mysql_errno\(\)](#) et [mysql_error\(\)](#) pour connaître la nature de l'erreur.

Exemple mysql_db_name()

```

<?php
error_reporting(E_ALL);
mysql_connect('dbhost', 'username', 'password');
$db_list = mysql_list_dbs();
$i = 0;
$cnt = mysql_num_rows($db_list);
while ($i < $cnt) {
echo mysql_db_name($db_list, $i) . "\n";
    $i++;
}
?>

```

Pour des raisons de compatibilité ascendante, mysql_dbname() est aussi accepté, mais obsolète.

10.45.8 mysql_db_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_db_query](#)(string *database*, string *query*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne un identifiant de résultat si la requête réussit, et FALSE sinon.

[mysql_db_query\(\)](#) Sélectionne une base de données et exécute une requête. Si l'identifiant de lien *link_identifier* n'est pas précisé, la fonction prendra par défaut la dernière base de données ouverte sur le

serveur, et si elle n'en trouve pas, elle tentera de se connecter, en utilisant la fonction [mysql_connect\(\)](#), sans arguments.

Voir aussi [mysql_connect\(\)](#).

Pour des raisons de compatibilité ascendante, `mysql()` peut aussi être utilisé.

[10.45.9 mysql_drop_db](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_drop_db](#)(string *database_name*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE sinon.

[mysql_drop_db\(\)](#) essaie d'effacer une base de données entière sur le serveur associé à l'identifiant de connexion *link_identifier*.

Voir aussi [mysql_create_db\(\)](#). Pour des raisons de comptabilité ascendante, `mysql_drop_db()` est toujours utilisable.

[10.45.10 mysql_errno](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_errno](#)(int *link_identifier*)

[PHP 3, PHP 4]

Les erreurs qui sont remontées depuis le serveur MySQL ne sont plus des alertes. A la place, il faut utiliser cette fonction pour obtenir le numéro d'erreur.

```
<?php
mysql_connect("marliesle");
echo mysql_errno().": ".mysql_error()."<BR>";
mysql_select_db("nonexistentdb");
echo mysql_errno().": ".mysql_error()."<BR>";
$conn = mysql_query("SELECT * FROM nonexistenttable");
echo mysql_errno().": ".mysql_error()."<BR>";
?>
```

Voir aussi [mysql_error\(\)](#).

[10.45.11 mysql_error](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_error](#)(int *link_identifier*)

[PHP 3, PHP 4]

Les erreurs générées par MySQL ne se transforment plus en alerte. A la place, elles sont accessibles via ces fonctions :

```
<?php
mysql_connect("marliesle");
echo mysql_errno().": ".mysql_error()."<BR>";
```

```
mysql_select_db("nonexistentdb");
echo mysql_errno().": ".mysql_error()."<BR>";
$conn = mysql_query("SELECT * FROM nonexistenttable");
echo mysql_errno().": ".mysql_error()."<BR>";
?>
```

Voir aussi [mysql_errno\(\)](#).

10.45.12 [mysql_fetch_array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_array](#)(int *result*, int *result_type*)
[PHP 3, PHP 4]

Retourne un tableau qui contient la ligne demandée, ou FALSE si il ne reste plus de ligne.

[mysql_fetch_array\(\)](#) est une version étendue de [mysql_fetch_row\(\)](#). En plus d'enregistrer les données sous forme d'un tableau à indice numérique, elle peut aussi les enregistrer dans un tableau associatif, en utilisant les noms des champs comme indices.

Si plusieurs colonnes ont le même nom, la dernière colonne aura la priorité. Pour accéder aux autres colonnes du même nom, vous devez utiliser l'index numériques, ou faire un alias pour chaque colonne.

```
select t1.f1 as foo t2.f1 as bar from t1, t2
```

Il est important de souligner que [mysql_fetch_array\(\)](#) N'est PAS plus lente que [mysql_fetch_row\(\)](#), tandis qu'elle ajoute un confort d'utilisation notable.

L'option *result_type* de [mysql_fetch_array\(\)](#) est une constant qui peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

Pour plus de détails, voir aussi [mysql_fetch_row\(\)](#).

mysql_fetch_array

```
<?php
mysql_connect($host,$user,$password);
$result = mysql_db_query("database","select * from table");
while($row = mysql_fetch_array($result)) {
echo $row["user_id"];
echo $row["fullname"];
}
mysql_free_result($result);
?>
```

10.45.13 [mysql_fetch_assoc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_assoc](#)(int *result*)
[PHP 4 >= 4.0.3]

[mysql_fetch_assoc\(\)](#) retourne un tableau associatif qui contient la ligne lue, ou bien FALSE, si il ne reste plus de lignes.

[mysql_fetch_assoc\(\)](#) est équivalente à [mysql_fetch_array\(\)](#) utilisée avec l'option MYSQL_ASSOC. Elle ne

retourne qu'un tableau associatif. C'est le fonctionnement original de [mysql_fetch_array\(\)](#). Si vous avez besoin d'indices numériques, utilisez [mysql_fetch_array\(\)](#).

Si plusieurs colonnes ont le même nom, la dernière aura la priorité. Pour accéder aux autres colonnes du même nom, vous devez utiliser [mysql_fetch_array\(\)](#) et les indices numériques.

Une chose importante à noter est que [mysql_fetch_assoc\(\)](#) n'est PAS significativement plus lente que [mysql_fetch_row\(\)](#), alors qu'elle apporte un confort d'utilisation important.

Pour plus de détails, reportez vous à [mysql_fetch_row\(\)](#) et [mysql_fetch_array\(\)](#).

mysql_fetch_assoc()

```
<?php
mysql_connect ($host, $user, $password);
$result = mysql_db_query ("database","select * from table");
while ($row = mysql_fetch_assoc ($result)) {
echo $row["user_id"];
echo $row["fullname"];
}
mysql_free_result ($result);
?>
```

10.45.14 mysql_fetch_field

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mysql_fetch_field](#) (int *result*, int *field_offset*)

[PHP 3, PHP 4]

Retourne un objet contenant les données.

[mysql_fetch_field\(\)](#) sert à obtenir des informations à propos des champs, dans certaines requêtes. Si l'offset du champs n'est pas spécifié, le champs suivant le dernier champs retourné, est retourné.

Les propriétés de l'objet sont :

- name – nom de la colonne
- table – nom de la table de la colonne
- max_length – taille maximale de la colonne
- not_null – 1 si la colonne ne peut pas être NULL (attribut NOT NULL)
- primary_key – 1 si la colonne est une clé primaire (attribut PRIMARY KEY)
- unique_key – 1 si la colonne est une clé unique (attribut UNIQUE)
- multiple_key – 1 si la colonne est une clé non-unique
- numeric – 1 si la colonne est numérique
- blob – 1 si la colonne est BLOB
- type – le type de la colonne
- unsigned – 1 si la colonne est non signé
- zerofill – 1 si la colonne est complétée par des zéros.

Voir aussi [mysql_field_seek\(\)](#).

10.45.15 mysql_fetch_lengths

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_lengths](#) (int *result*)

[PHP 3, PHP 4]

Retourne un tableau avec la taille de chaque colonne de la dernière ligne retournée par [mysql_fetch_row\(\)](#), sinon FALSE.

[mysql_fetch_lengths\(\)](#) stocke les tailles de chaque colonne de la dernière ligne retournée par [mysql_fetch_row\(\)](#), [mysql_fetch_array\(\)](#), et [mysql_fetch_object\(\)](#) dans un tableau, en commençant à la position.

Voir aussi [mysql_fetch_row\(\)](#).

10.45.16 mysql_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [mysql_fetch_object](#) (int *result*, int *result_typ*)

[PHP 3, PHP 4]

Retourne un objet dont les propriétés correspondent à une ligne d'un résultat, ou FALSE si il n'y a plus d'autres lignes.

[mysql_fetch_object\(\)](#) est identique à [mysql_fetch_array\(\)](#), à la différence qu'elle retourne un objet à la place d'un tableau. Vous pourrez ainsi accéder aux valeurs des champs par leur nom, mais plus par leur offset (les nombres ne sont pas des noms MySQL).

L'argument optionnel *result_typ* est une constante qui peut prendre les valeurs suivantes : MYSQL_ASSOC, MYSQL_NUM, et MYSQL_BOTH.

Concernant la vitesse, cette fonction est aussi rapide que [mysql_fetch_array\(\)](#), et presque aussi rapide que [mysql_fetch_row\(\)](#) (la différence est insignifiante)

mysql_fetch_object

```
<?php
mysql_connect($host,$user,$password);
$result = mysql_db_query("database","select * from table");
while($row = mysql_fetch_object($result)) {
    echo $row->user_id;
    echo $row->fullname;
}
mysql_free_result($result);
?>
```

Voir aussi [mysql_fetch_array\(\)](#) et [mysql_fetch_row\(\)](#).

10.45.17 mysql_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [mysql_fetch_row](#) (int *result*)

[PHP 3, PHP 4]

Retourne un tableau énuméré qui correspond à la ligne demandée, ou FALSE si il ne reste plus de ligne.

[mysql_fetch_row\(\)](#) va rechercher une ligne dans le résultat associé à l'identifiant de résultat spécifié. La ligne est retournée sous la forme d'un tableau. Chaque colonne est enregistré sous la forme d'un tableau commençant à la position 0.

Les appels suivants à [mysql_fetch_row\(\)](#) retourneront la ligne suivante dans le résultat, ou FALSE si il n'y a plus de ligne disponible.

Voir aussi [mysql_fetch_array\(\)](#), [mysql_fetch_object\(\)](#), [mysql_data_seek\(\)](#), [mysql_fetch_lengths\(\)](#), et [mysql_result\(\)](#).

10.45.18 mysql_field_flags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_flags](#) (int *result*, int *field_offset*)
[PHP 3, PHP 4]

[mysql_field_flags\(\)](#) retourne le sémaphore associé au champs spécifié par *field_offset*. Les sémaphores sont retournés comme des mots, séparés par des espaces, ce qui les rend facile à séparer, avec la commande [explode\(\)](#).

Les valeurs suivantes (pour une version suffisamment récente de MySQL) sont disponibles : "not_null", "primary_key", "unique_key", "multiple_key", "blob", "unsigned", "zerofill", "binary", "enum", "auto_increment", "timestamp".

Pour des raisons de compatibilité ascendante, [mysql_fieldflags\(\)](#) peut encore être utilisé.

10.45.19 mysql_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_name](#) (int *result*, int *field_index*)
[PHP 3, PHP 4]

[mysql_field_name\(\)](#) retourne le nom d'une colonne. Les arguments de la fonction sont un identifiant de résultat *result* et l'index du champs, ie. [mysql_field_name\(\\$result,2\)](#);

Retournera le nom du deuxième champs dans le résultat associé à \$result.

Pour des raisons de compatibilité ascendante, [mysql_fieldname\(\)](#) peut encore être utilisé.

10.45.20 mysql_field_len

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_field_len](#) (int *result*, int *field_offset*)
[PHP 3, PHP 4]

[mysql_field_len\(\)](#) retourne la taille du champs spécifié.

Pour des raisons de compatibilité ascendante, [mysql_fieldlen\(\)](#) peut encore être utilisé.

10.45.21 mysql_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_field_seek](#) (int *result*, int *field_offset*)
[PHP 3, PHP 4]

Place le pointeur de résultat sur le champs spécifié. Lors du prochain appel à [mysql_fetch_field\(\)](#) qui n'aura pas d'argument d'index de champs, le champs désormais pointé sera retourné.

Voir aussi [mysql_fetch_field\(\)](#).

10.45.22 mysql_field_table

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_table](#)(int *result*, int *field_offset*)

[PHP 3, PHP 4]

Retourne le nom de la table où se trouve une colonne. Pour des raisons de compatibilité ascendante, `mysql_fieldtable()` peut encore être utilisé.

10.45.23 mysql_field_type

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_field_type](#)(int *result*, int *field_offset*)

[PHP 3, PHP 4]

[mysql_field_type\(\)](#) est similaire à la fonction [mysql_field_name\(\)](#). Les arguments sont identiques, mais c'est le type du champs qui est retourné. Il vaudra "int", "real", "string", "blob", ou d'autres, comme détaillé dans la documentation MySQL.

Types mysql field

```

<?php
mysql_connect("localhost:3306");
mysql_select_db("wisconsin");
$result = mysql_query("SELECT * FROM onek");
$fields = mysql_num_fields($result);
$rows   = mysql_num_rows($result);
$i = 0;
$table = mysql_field_table($result, $i);
echo "Your '". $table. "' table has ". $fields. " fields et ". $rows. " records <BR>";
echo "The table has the following fields <BR>";
while ($i < $fields) {
    $type = mysql_field_type($result, $i);
    $name = mysql_field_name($result, $i);
    $len  = mysql_field_len($result, $i);
    $flags = mysql_field_flags($result, $i);
    echo $type. " ". $name. " ". $len. " ". $flags. "<BR>";
    $i++;
}
mysql_close();
?>

```

Pour des raisons de compatibilité ascendante, `mysql_fieldtype()` peut encore être utilisé.

10.45.24 mysql_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_free_result](#)(int *result*)

[PHP 3, PHP 4]

[mysql_free_result\(\)](#) n'est à appeler que si vous avez peur d'utiliser trop de mémoire durant l'exécution de votre script. Toute la mémoire associée à l'identifiant de résultat sera automatiquement libérée.

Pour des raisons de compatibilité ascendante, `mysql_freeresult()` peut encore être utilisé.

[10.45.25 mysql_insert_id](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_insert_id](#) (int *link_identifieur*)

[PHP 3, PHP 4]

[mysql_insert_id\(\)](#) retourne le dernier identifiant généré par un champs de type AUTO_INCREMENTED. Cette fonction ne prend aucun argument. Elle retourne le dernier identifiant généré par la dernière fonction INSERT effectuée.

[10.45.26 mysql_list_dbs](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_dbs](#) (int *link_identifieur*)

[PHP 3, PHP 4]

[mysql_list_dbs\(\)](#) retournera un identifiant de résultat, qui contiendra les noms des bases de données disponibles sur le serveur MySQL. Utilisez la fonction [mysql_tablename\(\)](#) pour lire toutes les bases de données.

Pour des raisons de compatibilité ascendante, `mysql_listdbs()` est encore disponible.

[10.45.27 mysql_list_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_fields](#) (string *database_name*, string *table_name*, int *link_identifieur*)

[PHP 3, PHP 4]

[mysql_list_fields\(\)](#) recherche les informations à propos de la table spécifiée. Les arguments sont la base de données, et le nom de la table. Un pointeur de résultat est retourné, et pourra être passé à [mysql_field_flags\(\)](#), [mysql_field_len\(\)](#), [mysql_field_name\(\)](#) et [mysql_field_type\(\)](#).

Un identifiant de résultat est un entier positif. La fonction retourne -1 si une erreur survient. Une chaîne décrivant le problème rencontré sera placée dans la variable `$phperrormsg`, et, à moins que la fonction n'ait été appelée sous la forme `@mysql()`, cette erreur sera aussi affichée.

Pour des raisons de compatibilité ascendante, `mysql_listfields()` est encore disponible.

[10.45.28 mysql_list_tables](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_list_tables](#) (string *database*, int *link_identifieur*)

[PHP 3, PHP 4]

[mysql_list_tables\(\)](#) prend le nom d'une base de données comme argument, et retourne un identifiant de résultat, qui contiendra la liste des tables. La fonction [mysql_tablename\(\)](#) est le meilleur moyen d'extraire les noms des tables depuis l'identifiant de résultat.

Pour des raisons de compatibilité ascendante, `mysql_listtables()` est encore disponible.

10.45.29 mysql_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_fields](#)(int *result*)

[PHP 3, PHP 4]

[mysql_num_fields\(\)](#) retourne le nombre de champs d'un résultat.

Voir aussi [mysql_db_query\(\)](#), [mysql_query\(\)](#), [mysql_fetch_field\(\)](#), [mysql_num_rows\(\)](#).

Pour des raisons de compatibilité ascendante [mysql_numfields\(\)](#) est encore disponible.

10.45.30 mysql_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_num_rows](#)(int *result*)

[PHP 3, PHP 4]

[mysql_num_rows\(\)](#) retourne le nombre de lignes d'un résultat. Cette commande n'est valide que pour les commandes SELECT . Pour connaître le nombre de lignes retournées par INSERT, UPDATE ou DELETE, utilisez [mysql_affected_rows\(\)](#).

Exemple mysql_num_rows() par crubel@trilizio.org

```
<?php
$conn = mysql_connect("adresse de l'hote", "utilisateur", "mot de passe");
mysql_select_db("base",$conn); // nécessaire si vous avez plusieurs bases
$resultfornummembers = mysql_query("SELECT * FROM Accounts",$conn);
$numMembers = mysql_num_rows($resultfornummembers);
echo "$numMembers Membres";
?>
```

Voir aussi [mysql_db_query\(\)](#), [mysql_query\(\)](#) et [mysql_fetch_row\(\)](#).

Pour des raisons de compatibilité ascendante [mysql_numrows\(\)](#) est encore disponible.

10.45.31 mysql_pconnect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_pconnect](#)(string *hostname :port :/path/to/socket* , string *username* , string *password*)

[PHP 3, PHP 4]

Retourne un lien persistant positif en cas de succès, et sinon FALSE en cas d'erreur.

[mysql_pconnect\(\)](#) établit une connexion persistante à un serveur MySQL. Tous les arguments sont optionnels, et des valeurs par défaut seront utilisés en cas d'omission ('localhost', nom d'utilisateur propriétaire du processus, mot de passe vide).

Le nom de l'hôte peut aussi inclure le numéro de port, c'est à dire "hostname:port" ou un chemin jusqu'à la socket ":/path/to/socket" pour l'hôte local. Note : *Le support de " :port" a été ajouté à partir de la version 3.0B4.*

Le support de " :/path/to/socket" a été ajouté à partir de la version 3.0.10.

[mysql_pconnect\(\)](#) se comporte exactement comme [mysql_connect\(\)](#), mais avec deux différences majeures :
Premièrement, lors de la connexion, la fonction essaie de trouver une connexion permanente déjà ouverte sur cet hôte, avec le même nom d'utilisateur et de mot de passe. Si une telle connexion est trouvée, son identifiant est retourné, sans ouvrir de nouvelle connexion.

Deuxièmement, la connexion au serveur MySQL ne sera pas terminée avec la fin du script. Au lieu de cela, le lien sera conservé pour un prochain accès ([mysql_close\(\)](#) ne terminera pas une connexion persistante établie par [mysql_pconnect\(\)](#)).

C'est pourquoi ce type de connexion est dite 'persistante'.

10.45.32 [mysql_query](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_query](#) (string *query*, int *link_identifier*)

[PHP 3, PHP 4]

[mysql_query\(\)](#) envoie une requête SQL à la base de données actuellement active sur le serveur MySQL. Si *link_identifier* n'est pas précisé, la dernière connexion est utilisée. Si aucune connexion n'a été ouverte, la fonction tentera d'en ouvrir une, avec la fonction [mysql_connect\(\)](#) mais sans aucun paramètre (c'est à dire avec les valeurs par défaut).

[mysql_connect\(\)](#)

[mysql_query\(\)](#) retourne TRUE ou FALSE, pour indiquer le succès ou l'échec de la requête. En cas de retour TRUE, la requête était valide et a pu être exécuté sur le serveur. Cela n'indique pas le nombre de ligne affectées, ou retournées. Il est parfaitement possible qu'une requête valide n'affecte aucune ligne ou ne retourne aucune ligne.

L'exemple suivant est syntaxiquement invalide, ce qui conduit [mysql_query\(\)](#) à l'échec, et retourne FALSE:

mysql_query()

```
<?php
$result = mysql_query ("SELECT * WHERE 1=1")
or die ("Invalid query");
?>
```

L'exemple suivant est sémantiquement invalide si *my_col* n'est pas une colonne de la table *my_tbl*, ce qui conduit [mysql_query\(\)](#) à l'échec, et retourne FALSE :

mysql_query()

```
<?php
$result = mysql_query ("SELECT my_col FROM my_tbl")
or die ("Invalid query");
?>
```

[mysql_query\(\)](#) échouera aussi et retournera aussi FALSE si les droits d'accès ne sont pas suffisants.

En supposant que la requête réussisse, vous pouvez appeler [mysql_affected_rows\(\)](#) pour connaître le nombre de lignes affectées (pour les commandes DELETE, INSERT, REPLACE, ou UPDATE). Pour les commandes SELECT , [mysql_query\(\)](#) retourne un identifiant de résultat que vous pouvez passer à [mysql_result\(\)](#). Lorsque vous avez terminé avec le résultat, libérez la mémoire avec [mysql_free_result\(\)](#).

Voir aussi [mysql_affected_rows\(\)](#), [mysql_db_query\(\)](#), [mysql_free_result\(\)](#), [mysql_result\(\)](#), [mysql_select_db\(\)](#) et [mysql_connect\(\)](#).

10.45.33 mysql_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_result](#) (int *result*, int *row*, mixed *field*)

[PHP 3, PHP 4]

[mysql_result\(\)](#) retourne le contenu d'un champs dans un résultat MySQL. L'argument de champs *row* peut être un offset de champs, ou le nom du champs, ou le nom de la table + point + le nom du champs (table.champs). Si la colonne a été aliasée, utilisez de préférence l'alias.

Lorsque vous travaillez sur des résultats de grande taille, vous devriez utiliser une des fonctions qui vont rechercher une ligne entière dans un tableau. Ces fonctions sont NETTEMENT plus rapides. De plus, l'utilisation d'un offset numériques est aussi beaucoup plus rapide que de spécifier un nom littéral.

Les appels [mysql_result\(\)](#) ne devraient pas être mélangés avec d'autres fonctions qui travaillent aussi sur le résultat.

Alternatives à haut rendement, RECOMMANDÉES : [mysql_fetch_row\(\)](#), [mysql_fetch_array\(\)](#) et [mysql_fetch_object\(\)](#).

10.45.34 mysql_select_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [mysql_select_db](#) (string *database_name*, int *link_identifiant*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, FALSE sinon.

[mysql_select_db\(\)](#) change la base de données active sur la connexion représentée par l'identifiant de connexion. Si aucun identifiant n'est spécifié, la dernière connexion est utilisée. S'il n'y a pas de dernière connexion, la fonction tentera de se connecter seule, avec [mysql_connect\(\)](#) et les paramètres par défaut.

Toutes les requêtes suivantes avec [mysql_query\(\)](#) seront faites avec la base de données active.

Voir aussi [mysql_connect\(\)](#), [mysql_pconnect\(\)](#), et [mysql_query\(\)](#).

Pour des raisons de compatibilité ascendante [mysql_selectdb\(\)](#) est encore disponible.

10.45.35 mysql_tablename

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [mysql_tablename](#) (int *result*, int *i*)

[PHP 3, PHP 4]

[mysql_tablename\(\)](#) prend le pointeur de résultat obtenu avec [mysql_list_tables\(\)](#) ou bien un index entier, et retourne le nom de la table. La fonction [mysql_num_rows\(\)](#) peut être utilisée pour déterminer le nombre de tables dans le pointeur de résultat.

Exemple [mysql_tablename\(\)](#)

```
<?php
mysql_connect ("localhost:3306");
$result = mysql_list_tables ("wisconsin");
$i = 0;
while ($i < mysql_num_rows ($result)) {
    $tb_names[$i] = mysql_tablename ($result, $i);
    echo $tb_names[$i] . "<BR>";
    $i++;
}
```

```
}
?>
```

10.46 Réseau

[\[Notes en ligne\]](#)

10.46.1 checkdnsrr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [checkdnsrr](#)(string *host*, string *type*)
[PHP 3, PHP 4]

[checkdnsrr\(\)](#) recherche l'enregistrement DNS de type *type* correspondant à l'hôte *host*. Retourne TRUE si un record a été trouvé, et FALSE en cas d'erreur ou d'échec.

type peut prendre les valeurs suivantes : A, MX, NS, SOA, PTR, CNAME, ou ANY. Par défaut, la valeur est : MX.

host peut être soit une adresse IP de la forme x.x.x.x avec x entre 0 et 256, soit un nom d'hôte.

Voir aussi [getmxrr\(\)](#), [gethostbyaddr\(\)](#), [gethostbyname\(\)](#), [gethostbynameal\(\)](#) et la page de manuel `named(8)`.

10.46.2 closelog

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [closelog](#)
[PHP 3, PHP 4]

[closelog\(\)](#) ferme le pointeur qui sert à écrire dans l'historique système. L'utilisation de [closelog\(\)](#) est optionnelle.

Voir aussi [define_syslog_variables\(\)](#), [syslog\(\)](#) et [openlog\(\)](#).

10.46.3 debugger off

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [debugger_off](#)
[PHP 3]

[debugger_off\(\)](#) inactive le débogueur interne de PHP. Le débogueur est toujours en cours de développement.

10.46.4 debugger on

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [debugger_on](#)(string *address*)
[PHP 3]

[debugger_on\(\)](#) active le débogueur interne de PHP, et le connecte à l'adresse *address*. Le débogueur est toujours en cours de développement.

10.46.5 define_syslog_variables

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [define_syslog_variables](#)(void)

[PHP 3, PHP 4]

[define_syslog_variables\(\)](#) initialise toutes les constantes utilisées par les fonctions de syslog. Voir aussi [openlog\(\)](#), [syslog\(\)](#) et [closelog\(\)](#).

10.46.6 fsockopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [fsockopen](#)(string *udp://hostname* , int *port* , int *errno* , string *errstr* , double *timeout*)

[PHP 3, PHP 4]

[fsockopen\(\)](#) créer un flot de connexion à l'Internet (AF_INET) ou à un domaine Unix (AF_UNIX). Via Internet, cette fonction va ouvrir une socket de connexion TCP avec l'hôte *hostname* sur le port *port*. Pour les connexions UDP, vous devez explicitement spécifier le protocole : *udp://hostname*. Via un domaine Unix, *hostname* représente le chemin jusqu'à la socket, et *port* doit être mis à 0. L'option *timeout* sert à donner une durée maximale à cet appel.

[fsockopen\(\)](#) retourne un pointeur de fichier qui peut être utilisé avec d'autres fonctions fichiers, telles que [fgets\(\)](#), [fgetss\(\)](#), [fputs\(\)](#), [fclose\(\)](#) et [feof\(\)](#).

Si l'appel échoue, [fsockopen\(\)](#) retourne FALSE, et si les options *errno* et *errstr* ont été fournies, elles contiennent désormais les raisons de l'échec. Si l'erreur retournée est 0 et que la fonction retourne FALSE, c'est une indication d'erreur. C'est probablement dû à une erreur d'initialisation de la socket. Notez que *errno* et *errstr* sont passées par référence.

Suivant les environnements, le type 'domaine Unix' ou l'option *timeout* ne sont pas toujours disponibles.

La socket sera ouverte par défaut en mode bloquant. Vous pouvez changer de mode en utilisant :

[socket_set_blocking\(\)](#).

Exemple avec fsockopen()

```
<?php
$fp = fsockopen("www.php.net", 80, &$errno, &$errstr, 30);
if(!$fp) {
    echo "$errstr ($errno)<br>\n";
} else {
    fputs($fp, "GET / HTTP/1.0\n\n");
    while(!feof($fp)) {
        echo fgets($fp,128);
    }
    fclose($fp);
}
?>
```

L'exemple ci-dessous décrit comment lire la date et l'heure grâce à un service UDP "daytime" (port 13), sur votre propre machine.

Utilisation d'une connexion UDP

```
<?php
$fp = fsockopen("udp://127.0.0.1", 13, &$errno, &$errstr);
if (!$fp) {
```

```

echo "ERREUR: $errno - $errstr<br>\n";
} else {
fwrite($fp, "\n");
echo fread($fp, 26);
fclose($fp);
}
?>

```

Note : Le paramètre *timeout* a été introduit en PHP 3.0.9 et le support UDP en PHP 4.

Voir aussi: [pfsockopen\(\)](#), [socket_set_blocking\(\)](#), [socket_set_timeout\(\)](#), [fgets\(\)](#), [fgetss\(\)](#), [fputs\(\)](#), [fclose\(\)](#), et [feof\(\)](#).

10.46.7 gethostbyaddr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gethostbyaddr](#) (string *ip_address*)
[PHP 3, PHP 4]

[gethostbyaddr\(\)](#) retourne le nom d'hôte correspondant à l'IP *ip_address*. Si une erreur survient, retourne *ip_address*.

Voir aussi [gethostbyname\(\)](#).

10.46.8 gethostbyname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gethostbyname](#) (string *hostname*)
[PHP 3, PHP 4]

[gethostbyname\(\)](#) retourne l'adresse IP correspondant à l'hôte *hostname*.

Voir aussi [gethostbyaddr\(\)](#).

10.46.9 gethostbynameel

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gethostbynameel](#) (string *hostname*)
[PHP 3, PHP 4]

[gethostbynameel\(\)](#) retourne la liste d'IP correspondant à l'hôte *hostname*.

Voir aussi [gethostbyname\(\)](#), [gethostbyaddr\(\)](#), [checkdnsrr\(\)](#), [getmxrr\(\)](#) et la page 8 du manuel.

10.46.10 getmxrr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getmxrr](#) (string *hostname*, array *mxhosts*, array *weight*)
[PHP 3, PHP 4]

[getmxrr\(\)](#) effectue une recherche DNS pour obtenir les enregistrements MX de l'hôte *hostname*. Retourne TRUE si des enregistrements sont trouvés, et FALSE si une erreur est rencontrée, ou si la recherche échoue. La liste des enregistrements MX est placée dans le tableau *mxhosts*. Si le tableau *weight* est fourni, il sera

rempli par les informations de poids.

Voir aussi [checkdnsrr\(\)](#), [gethostbyname\(\)](#), [gethostbyname_l\(\)](#), [gethostbyaddr\(\)](#), et la page 8 du manuel.

10.46.11 getprotobyname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getprotobyname](#)(string *name*)

[PHP 4 >= 4.0b4]

[getprotobyname\(\)](#) retourne le numéro de protocole associé avec le nom de protocole *name*, comme dans ``/etc/protocols'`.

Voir aussi [getprotobynumber\(\)](#).

10.46.12 getprotobynumber

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getprotobynumber](#)(int *number*)

[PHP 4 >= 4.0b4]

[getprotobynumber\(\)](#) retourne le numéro de protocole associé avec le nom de protocole *name*, comme dans ``/etc/protocols'`.

Voir aussi [getprotobyname\(\)](#).

10.46.13 getservbyname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [getservbyname](#)(string *service*, string *protocol*)

[PHP 4 >= 4.0b4]

[getservbyname\(\)](#) retourne le numéro de port associé à au service *service* et protocole *protocol*, comme dans ``/etc/services'`. *protocol* vaut soit tcp ou udp.

Voir aussi [getservbyport\(\)](#).

10.46.14 getservbyport

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [getservbyport](#)(int *port*, string *protocol*)

[PHP 4 >= 4.0b4]

[getservbyport\(\)](#) le service internet associé au port *port* pour le protocole *protocol* comme dans ``/etc/services'`. *protocol* vaut soit tcp ou udp.

Voir aussi [getservbyname\(\)](#).

10.46.15 ip2long

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ip2long](#)(string *ip_address*)

[PHP 4 >= 4.0RC1]

[ip2long\(\)](#) génère une adresse IPv4 à partir de son équivalent numérique.**Exemple [ip2long\(\)](#)**

```
<?php
$ip = gethostbyname("www.php.net");
$out = "Les URLs suivantes sont équivalentes :<br>\n";
$out .= "http://www.php.net/, http://".$ip.", et http://".ip2long($ip)."/<br>\n";
echo $out;
?>
```

Voir aussi: [long2ip\(\)](#)

[10.46.16 long2ip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)string [long2ip](#)(int *proper_address*)

[PHP 4 >= 4.0RC1]

[long2ip\(\)](#) génère une adresse IP (format aaa.bbb.ccc.ddd) à partir de sa représentation littérale.Voir aussi: [ip2long\(\)](#)

[10.46.17 openlog](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)int [openlog](#)(string *ident*, int *option*, int *facility*)

[PHP 3, PHP 4]

[openlog\(\)](#) ouvre la connexion à l'historique système. La chaîne *ident* sera ajouté à chaque message. Les valeurs de *option* et *facility* sont données ci-dessous. L'utilisation de [openlog\(\)](#) est optionnelle; cette fonction sera automatiquement appelée par [syslog\(\)](#) si nécessaire, et dans ce cas, l'identification sera mise par défaut à FALSE. *facility* sert à indiquer quel programme enregistre ce message. Cela vous permet de spécifier (sur la machine d'historique) comment traiter les messages venant de plusieurs serveurs.

Constante	Description
LOG_CONS	Si une erreur survient lors de l'envoi des données au gestionnaire d'historique, écrire directement l'erreur sur la console. @tab
LOG_NDELAY	Ouvre immédiatement une connexion au gestionnaire d'historique @tab
LOG_ODELAY	(par défaut) retarde l'ouverture de la connexion jusqu'à ce que le premier message soit enregistré @tab
LOG_PERROR	Envoie le message au gestionnaire standard

LOG_PID	Inclus le PID à chaque message
---------	--------------------------------

Vous pouvez utiliser une ou plusieurs de ces options. Pour les combiner, utiliser l'opérateur OR. Par exemple, pour ouvrir immédiatement la connexion, écrire sur la console et inclure le PID de chaque message, utilisez : LOG_CONS | LOG_NDELAY | LOG_PID.

Constante	Description
LOG_AUTH	securité/messages d'autorisation (utilisez LOG_AUTHPRIV, pour remplacer cette constante sur les systèmes où elle est définie). @tab
LOG_AUTHPRIV	securité/messages d'autorisation (privé)
LOG_CRON	démon horloge (cron et at)
LOG_DAEMON	autres démons système
LOG_KERN	noyau (kernel)
LOG_LOCAL0 ... LOG_LOCAL7	reservé pour utilisation ultérieure
LOG_LPR	imprimante (line printer subsystem)
LOG_MAIL	messagerie mail
LOG_NEWS	USENET newsgroup
LOG_SYSLOG	messages generé en interne par syslogd
LOG_USER	messages utilisateurs générique
LOG_UUCP	UUCP subsystem

Voir aussi [define_syslog_variables\(\)](#), [syslog\(\)](#) et [closelog\(\)](#).

10.46.18 pfsockopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pfsockopen](#) (string *hostname*, int *port*, int *errno* , string *errstr* , int *timeout*)
[PHP 3>= 3.0.7, PHP 4]

[pfsockopen\(\)](#) se comporte exactement comme [fsockopen\(\)](#) mais la connexion ouverte le reste, même après la fin du script. C'est la version persistante de [fsockopen\(\)](#).

10.46.19 socket_get_status

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [socket_get_status](#) (resource *socket_get_status*)
[PHP 4 >= 4.0b4]

[socket_get_status\(\)](#) retourne les informations sur la socket *socket_get_status*, et fournit la réponse sous la forme d'un tableau à quatre entrées:

- ***timed_out*** (bool) – La socket a expirée en attendant des données
- ***blocked*** (bool) – La socket a été bloquée
- ***eof*** (bool) – Indique un événement fin de fichier (EOF)
- ***unread_bytes*** (int) – Nombre d'octets restant dans les buffers de la socket.

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#) et [strerror\(\)](#).

10.46.20 socket_set_blocking

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [socket_set_blocking](#)(int *socket descriptor*, int *mode*)

[PHP 4 >= 4.0b4]

Si *mode* est FALSE, la socket est mise en mode non bloquant, et si il est TRUE, la socket est mise en mode bloquant. Cela affecte des appels tels que [fgets\(\)](#) qui lisent depuis une socket. En mode non bloquant, un appel [fgets\(\)](#) retournera immédiatement toujours TRUE tandis qu'en mode bloquant, elle va attendre que des données arrivent pour répondre TRUE.

10.46.21 socket_set_timeout

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [socket_set_timeout](#)(int *socket descriptor*, int *seconds*, int *microseconds*)

[PHP 4 >= 4.0b4]

[socket_set_timeout\(\)](#) fixe la durée de vie de la socket *socket descriptor*, exprimée comme la somme de *seconds* secondes et *microseconds* micro-secondes.

Exemple [socket_set_timeout\(\)](#)

```
<?php
$fp = fsockopen("http://www.php.net", 80);
if(!$fp) {
    echo "Unable to open\n";
} else {
    fputs($fp, "GET / HTTP/1.0\n\n");
    $start = time();
    socket_set_timeout($fp, 2);
    $res = fread($fp, 2000);
    var_dump(socket_get_status($fp));
    fclose($fp);
    print $res;
}
?>
```

Cette fonction s'appelait [set_socket_timeout\(\)](#) mais elle est désormais obsolète.

Voir aussi: [fsockopen\(\)](#) et [fopen\(\)](#).

10.46.22 syslog

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [syslog](#)(int *priority*, string *message*)

[PHP 3, PHP 4]

[syslog\(\)](#) génère un message qui sera inscrit dans l'historique par le système. **priority** est une combinaison des valeurs d'accès et de niveau, qui seront décrites dans la prochaine section. Les derniers arguments sont le message à envoyer. Attention : les caractères %m seront remplacés par l'erreur (sous forme de chaîne), présente dans `errno`.

Constante	Description
LOG_EMERG	système inutilisable
LOG_ALERT	une décision doit être prise immédiatement
LOG_CRIT	conditions critiques
LOG_ERR	conditions d'erreur
LOG_WARNING	conditions d'alerte
LOG_NOTICE	condition normale, mais significative
LOG_INFO	message d'information
LOG_DEBUG	message de débogage

Utilisation de syslog()

```
<?php
define_syslog_variables();
// ouverture de syslog, ajout du PID et envoie simultané du
// message à la sortie standard et à un mecanisme
// spécifique
openlog("myScripLog", LOG_PID | LOG_PERROR, LOG_LOCAL0);
// quelques lignes de code
if (authorized_client()) {
    // faire quelquechose
} else {
    // client non autorisé!
    // notation de la tentative
    $access = date("Y/m/d H:i:s");
    syslog(LOG_WARNING, "Client non autorisé: $access $REMOTE_ADDR ($HTTP_USER_AGENT)");
}
closelog();
?>
```

Pour plus d'informations sur comment mettre en place un gestionnaire d'historique, reportez vous au manuel Unix, page 5 `syslog.conf` 5. D'autres informations sur les systèmes d'historique et leurs options sont aussi disponibles dans le manuel `syslog` 3 des machines Unix.

Avec Windows NT, l'historique est pris en charge par Event Log.

10.47 NIS

[\[Notes en ligne\]](#)

NIS (feu Yellow Pages / Pages jaunes) permet la gestion par le réseau de fichiers d'administration importants (tel un fichier de mot de passe). Pour plus d'informations, reportez vous au manuel NIS, ou à [Introduction to](#)

[YP/NIS](#) Introduction to YP/NIS (en anglais). Il existe un livre en anglais [Managing NFS and NIS](#) par Hal Stern.

Pour ajouter ces fonctionnalités, vous devez compiler PHP avec l'option `--with-yp`(PHP 3) ou `--enable-yp`(PHP 4).

[10.47.1 yp_get_default_domain](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yp_get_default_domain](#)(void)

[PHP 3>= 3.0.7, PHP 4]

[yp_get_default_domain\(\)](#) retourne le nom de domaine NIS par défaut. Ce nom de domaine peut être utilisé pour les futurs appels NIS.

Un domaine NIS peut être décrit comme un regroupement de cartes NIS. Tous les hôtes qui ont besoin d'informations, s'attachent à un domaine. Référez vous aux documents cités en début de document pour plus de détails.

Exemple avec le domaine par défaut

```
<?php
    $domain = yp_get_default_domain();
    echo "Le domaine par défaut est : " . $domain;
?>
```

[10.47.2 yp_order](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yp_order](#)(string *domain*, string *map*)

[PHP 3>= 3.0.7, PHP 4]

[yp_order\(\)](#) retourne le numéro d'ordre d'une carte ou FALSE.

Exemple d'ordre NIS

```
<?php
    $number = yp_order($domain,$mapname);
    echo "Le numéro d'ordre de cette carte est : " . $order;
?>
```

Voir aussi [yp_get_default_domain\(\)](#).

[10.47.3 yp_master](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [yp_master](#)(string *domain*, string *map*)

[PHP 3>= 3.0.7, PHP 4]

[yp_master\(\)](#) retourne le nom de la machine maître d'une carte.

Exemple de maître NIS

```
<?php
    $number = yp_master($domain, $mapname);
    echo "Master for this map is: " . $master;
?>
```

Voir aussi [yp-get-default-domain\(\)](#).

10.47.4 yp_match

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [yp_match](#)(string *domain*, string *map*, string *key*)
[PHP 3>= 3.0.7, PHP 4]

[yp_match\(\)](#) retourne la valeur associée à la clé passée en argument, pour la carte spécifiée, ou FALSE. La clé doit exister et être exacte.

Exemple de recherche NIS

```
<?php
    $entry = yp_match($domain, "passwd.byname", "joe");
    echo "La valeur trouvée est: " . $entry;
?>
```

Dans le cas présent, ce pourrait être: joe:##joe:11111:100:Joe User:/home/j/joe:/usr/local/bin/bash

Voir aussi [yp-get-default-domain\(\)](#)

10.47.5 yp_first

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [yp_first](#)(string *domain*, string *map*)
[PHP 3>= 3.0.7, PHP 4]

[yp_first\(\)](#) retourne le premier couple (clé ; valeur) d'une carte donnée, ou FALSE.

Exemple avec yp_first

```
<?php
    $entry = yp_first($domain, "passwd.byname");
    $key = key($entry);
    echo "La première entrée de cette carte est " . $key
        . " et sa valeur est " . $entry[$key];
?>
```

Voir aussi [yp-get-default-domain\(\)](#)

10.47.6 yp_next

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [yp_next](#)(string *domain*, string *map*, string *key*)

[PHP 3>= 3.0.7, PHP 4]

[yp_next\(\)](#) retourne le couple (clé ; valeur) suivant la clé donnée d'une carte donnée ou FALSE.

Exemple

```
<?php
    $entry = yp_next($domain, "passwd.byname", "joe");
    if(!$entry) {
        echo yp_errno() . ": " . yp_err_string();
    }
    $key = key($entry);
    echo "L'entree suivante après joe a la cle " . $key
        . " et sa valeur " . $entry[$key];
?>
```

Voir aussi [yp-get-default-domain\(\)](#).

10.48 Oracle 8

[\[Notes en ligne\]](#)

Ces fonctions vous permettront d'accéder aux serveurs Oracle8 et Oracle7. Elles utilisent l'interface Oracle8 Call-Interface (OCI8). Vous aurez donc besoin des librairies clientes Oracle8 pour pouvoir les utiliser. Il faut noter que cette extension est plus souple que l'extension Oracle officielle. Elle supporte notamment les liaisons entre les variables globales et locales de PHP avec des emplacements Oracle; elle supporte complètement les types LOB, FILE et ROWID et vous permet d'utiliser des variables de définitions personnalisables.

Avant d'utiliser cette extension, assurez vous que vous avez bien paramétré vos variables d'environnement Oracle, ainsi que votre démon utilisateur. Les variables dont vous pouvez avoir besoin sont :

- ORACLE_HOME
- ORACLE_SID
- LD_PRELOAD
- LD_LIBRARY_PATH
- NLS_LANG
- ORA_NLS33

Après avoir configuré ces variables pour votre utilisateur "serveur web", assurez vous aussi d'ajouter cet utilisateur (nobody, www) au group Oracle.

Aide OCI

```
<?php
// par sergo@bacup.ru
// Utilisez l'option : OCI_DEFAULT pour exécuter les commandes avec un délai
```



```

OCIExecute($stmt, OCI_DEFAULT);
// pour lire les données après lecture, utilisez :
$result = OCIResult($stmt, $n);
if (is_object ($result)) $result = $result->load();
// Pour les commandes INSERT ou UPDATE utilisez:
$sql = "insert into table (field1, field2) values (field1 = 'value',
field2 = empty_clob()) returning field2 into :field2";
OCIParse($conn, $sql);
$clob = OCINewDescriptor($conn, OCI_D_LOB);
OCIBindByName ($stmt, ":field2", &$clob, -1, OCI_B_CLOB);
OCIExecute($stmt, OCI_DEFAULT);
$clob->save ("some text");
?>

```

Vous pouvez facilement accéder aux procédures stockées, de la même façon que vous le feriez par ligne de commande :

Utilisation de procédures stockées

```

<?php
// par webmaster@remoterealty.com
$stmt = OCIExecute ( $dbh, "begin sp_newaddress( :address_id, '$firstname',
'$lastname', '$company', '$address1', '$address2', '$city', '$state',
'$postalcode', '$country', :error_code );end;" );
// Ce script appelle la procédure stockée sp_newaddress, avec address_id qui est
// une variable entrante/sortante et :error_code une variable sortante.
// Lorsque vous les liez :
OCIBindByName ( $sth, ":address_id", $addr_id, 10 );
OCIBindByName ( $sth, ":error_code", $errorcode, 10 );
OCIExecute ( $sth );
?>

```

10.48.1 ocidefinebyname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocidefinebyname\(\)](#) (int *stmt*, string *Column-Name*, mixed *variable*, int *type*)

[PHP 3>= 3.0.7, PHP 4]

[ocidefinebyname\(\)](#) copie les valeurs issues de colonnes SQL *Column-Name* dans les variables PHP.

Méfiez-vous des colonnes Oracle qui sont toutes en majuscule, tandis que dans les SELECT, vous pouvez aussi les écrire en minuscules. [ocidefinebyname\(\)](#) s'attend à ce que *Column-Name* soit en majuscules. Si vous définissez une variable qui n'existe pas dans la commande SELECT, vous ne serez pas prévenu par une erreur.

Si vous avez besoin de définir un type de données abstrait, tel que (LOB/ROWID/BFILE), vous devez lui allouer la mémoire avec [ocinewdescriptor\(\)](#). Reportez vous aussi à [ocibindbyname\(\)](#).

OCIDefineByName

```

<?php
/* Exemple OCIDefineByPos par thies@thieso.net (980219) */
$conn = OCILogon("scott","tiger");
$stmt = OCIExecute($conn,"select empno, ename from emp");
/* La définition DOIT être faite AVANT ociexecute! */
OCIDefineByName($stmt, "EMPNO", &$empno);

```

```

OCIDefineByName($stmt,"ENAME",&$ename);
OCIExecute($stmt);
while (OCIFetch($stmt)) {
    echo "empno:". $empno. "\n";
    echo "ename:". $ename. "\n";
}
OCIFreeStatement($stmt);
OCILogoff($conn);
?>

```

10.48.2 ocibindbyname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocibindbyname](#)(int *stmt*, string *ph_name*, mixed *&variable*, int *length*, int *type*)
[PHP 3>= 3.0.4, PHP 4]

[ocibindbyname\(\)](#) relie la variable PHP *variable* à l'emplacement Oracle *ph_name*. Son utilisation (comme entrée ou comme sortie) sera définie à l'exécution, et l'espace nécessaire sera alloué. Le paramètre de longueur *length* fixe la taille maximum pour la liaison. Si vous affectez une longueur de -1, [ocibindbyname\(\)](#) utilisera la longueur de variable comme maximum.

Si vous devez lier des types abstraits de données (LOB/ROWID/BFILE), vous devrez l'allouer dans un premier temps, avec [ocinewdescriptor\(\)](#). La longueur *length* ne sert pas pour ces types et devrait être fixée à -1. La variable *type* indique au serveur Oracle, quel type de pointeur va être utilisé. Les valeurs possibles sont : OCI_B_FILE (Fichier binaires), OCI_B_CFILE (Fichier texte), OCI_B_CLOB (LOB- texte), OCI_B_BLOB (LOB binaire) et OCI_B_ROWID (ROWID).

OCIDefineByName

```

<?php
/* Exemple OCIBindByPos par thies@thieso.net (980221)
Insère 3 lignes dans emp, et utilise ROWID pour mettre à jour
les lignes, juste après l'insertion.
*/
$conn = OCILogon("scott","tiger");
$stmt = OCIParse($conn,"insert into emp (empno, ename) ".
    "values (:empno,:ename) ".
    "returning ROWID into :rid");
$data = array(1111 => "Larry", 2222 => "Bill", 3333 => "Jim");
$rowid = OCINewDescriptor($conn,OCI_D_ROWID);
OCIBindByName($stmt,":empno",&$empno,32);
OCIBindByName($stmt,":ename",&$ename,32);
OCIBindByName($stmt,":rid",&$rowid,-1,OCI_B_ROWID);
$update = OCIParse($conn,"update emp set sal = :sal where ROWID = :rid");
OCIBindByName($update,":rid",&$rowid,-1,OCI_B_ROWID);
OCIBindByName($update,":sal",&$sal,32);
$sal = 10000;
while (list($empno,$ename) = each($data)) {
    OCIExecute($stmt);
    OCIExecute($update);
}
$rowid->free();
OCIFreeStatement($update);
OCIFreeStatement($stmt);
$stmt = OCIParse($conn,"select * from emp where empno in (1111,2222,3333)");
OCIExecute($stmt);
while (OCIFetchInto($stmt,&$arr,OCI_ASSOC)) {
    var_dump($arr);
}

```

```

}
OCIFreeStatement($stmt);
/* Effacement des lignes inutiles dans la table emp .... */
$stmt = OCIParse($conn,"delete from emp where empno in (1111,2222,3333)");
OCIExecute($stmt);
OCIFreeStatement($stmt);
OCILogoff($conn);
?>

```

10.48.3 ocilogon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocilogon\(\)](#)(string *username*, string *password*, string *db*)

[PHP 3>= 3.0.4, PHP 4]

[ocilogon\(\)](#) retourne un identifiant de connexion, nécessaire à la plus part des fonctions OCI. Si l'option ORACLE_SID n'est pas précisée, PHP utilisera la variable d'environnement ORACLE_SID pour déterminer le serveur de connexion.

Les connexions sont partagées, à l'intérieur d'une même page avec [ocilogon\(\)](#). Cela signifie que COMMIT et ROLLBACK s'appliquent à toutes les transactions commencées à l'intérieur d'une même page, même si vous avez créé de multiples connexions.

Cet exemple montre comment les connexions sont partagées :

OCILogon

```

<?php
print "<HTML><PRE>";
$db = "";
$c1 = ocilogon("scott","tiger",$db);
$c2 = ocilogon("scott","tiger",$db);
function create_table($conn)
{ $stmt = ociparse($conn,"create table scott.hallo (test
varchar2(64))");
ociexecute($stmt);
echo $conn." created table\n\n";
}
function drop_table($conn)
{ $stmt = ociparse($conn,"drop table scott.hallo");
ociexecute($stmt);
echo $conn." dropped table\n\n";
}
function insert_data($conn)
{ $stmt = ociparse($conn,"insert into scott.hallo values('$conn' || ' ' || to_char(sysdate,'DD-MO
ociexecute($stmt,OCI_DEFAULT);
echo $conn." inserted hallo\n\n";
}
function delete_data($conn)
{ $stmt = ociparse($conn,"delete from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn." deleted hallo\n\n";
}
function commit($conn)
{ ocicommit($conn);
echo $conn." committed\n\n";
}
function rollback($conn)
{ ocirollback($conn);
echo $conn." rollback\n\n";
}

```

```

function select_data($conn)
{ $stmt = ociparse($conn,"select * from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn."----selecting\n\n";
while (ocifetch($stmt))
echo $conn." <".ociresult($stmt,"TEST").">\n\n";
echo $conn."----done\n\n";
}
create_table($c1);
insert_data($c1);    // Insertion d'une ligne avec c1
insert_data($c2);    // Insertion d'une ligne avec c2
select_data($c1);    // Les résultats des deux insertions sont retournés
select_data($c2);
rollback($c1);       // Annulation avec c1
select_data($c1);    // Les résultats des deux insertions sont annulés
select_data($c2);
insert_data($c2);    // Insertion d'une ligne avec c2
commit($c2);         // Validation avec using c2
select_data($c1);    // Le résultat de c2 est retourné
delete_data($c1);    // Effacement de toutes les lignes avec c1
select_data($c1);    // Aucune ligne n'est retournée
select_data($c2);    // Aucune ligne n'est retournée
commit($c1);         // Validation avec c1
select_data($c1);    // Aucune ligne n'est retournée
select_data($c2);    // Aucune ligne n'est retournée
drop_table($c1);
print "</PRE></HTML>";
?>

```

Voir aussi [ociploton\(\)](#) et [ocinlogon\(\)](#).

10.48.4 ociploton

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ociploton](#) (string *username*, string *password*, string *db*)

[PHP 3>= 3.0.8, PHP 4]

[ociploton\(\)](#) crée une connexion persistante à un serveur Oracle 8 et s'authentifie. Si l'option ORACLE_SID n'est pas spécifiée, PHP utilisera la variable d'environnement ORACLE_SID pour déterminer le serveur de connexion.

Voir aussi [ociploton\(\)](#) et [ocinlogon\(\)](#).

10.48.5 ocinlogon

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocinlogon](#) (string *username*, string *password*, string *db*)

[PHP 3>= 3.0.8, PHP 4]

[ocinlogon\(\)](#) crée une nouvelle connexion à un serveur Oracle et s'authentifie. Si l'option ORACLE_SID n'est pas spécifié, PHP utilisera la variable d'environnement ORACLE_SID pour déterminer le serveur de connexion.

[ocinlogon\(\)](#) force le serveur à établir une nouvelle connexion. Cette fonction ne doit être utilisée que si vous voulez isoler un ensemble de transactions. Par défaut, les connexions sont partagées au niveau de la page, si

vous utilisez la fonction [ocinlogon\(\)](#) ou bien au niveau du processus web, si vous utilisez [ociplogon\(\)](#). Si vous avez de multiples connexions ouvertes avec [ocinlogon\(\)](#), les validations et annulations ne s'appliquent qu'à la connexion spécifiée.

L'exemple ci dessous montre l'utilisation des connexions séparées.

OCINLogon

```
<?php
print "<HTML><PRE>";
$db = "";
$c1 = ocilogon("scott","tiger",$db);
$c2 = ocinlogon("scott","tiger",$db);
function create_table($conn)
{ $stmt = ociparse($conn,"create table scott.hallo (test
varchar2(64))");
ociexecute($stmt);
echo $conn." created table\n\n";
}
function drop_table($conn)
{ $stmt = ociparse($conn,"drop table scott.hallo");
ociexecute($stmt);
echo $conn." dropped table\n\n";
}
function insert_data($conn)
{ $stmt = ociparse($conn,"insert into scott.hallo values('$conn' || ' ' || to_char(sysdate,'DD-MO
ociexecute($stmt,OCI_DEFAULT);
echo $conn." inserted hallo\n\n";
}
function delete_data($conn)
{ $stmt = ociparse($conn,"delete from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn." deleted hallo\n\n";
}
function commit($conn)
{ ocicommit($conn);
echo $conn." committed\n\n";
}
function rollback($conn)
{ ocirollback($conn);
echo $conn." rollback\n\n";
}
function select_data($conn)
{ $stmt = ociparse($conn,"select * from scott.hallo");
ociexecute($stmt,OCI_DEFAULT);
echo $conn."----selecting\n\n";
while (ocifetch($stmt))
echo $conn." <".ociresult($stmt,"TEST").">\n\n";
echo $conn."----done\n\n";
}
create_table($c1);
insert_data($c1);
select_data($c1);
select_data($c2);
rollback($c1);
select_data($c1);
select_data($c2);
insert_data($c2);
commit($c2);
select_data($c1);
delete_data($c1);
select_data($c1);
select_data($c2);
```

```

commit($c1);
select_data($c1);
select_data($c2);
drop_table($c1);
print "</PRE></HTML>";
?>

```

Voir aussi [ocilogon\(\)](#) et [ociplogon\(\)](#).

10.48.6 ocilogoff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocilogoff](#) (int *connection*)
 [PHP 3>= 3.0.4, PHP 4]

[ocilogoff\(\)](#) ferme la connexion Oracle.

10.48.7 ociexecute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ociexecute](#) (int *statement*, int *mode*)
 [PHP 3>= 3.0.4, PHP 4]

[ociexecute\(\)](#) exécute une commande déjà préparée (voir [ociparse\(\)](#)). L'option *mode* vous permet de spécifier le mode d'exécution (par défaut, il est à OCI_COMMIT_ON_SUCCESS). Si vous ne voulez pas que la commande soit automatiquement validée, utilisez le mode OCI_DEFAULT.

10.48.8 ocicommit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocicommit](#) (int *connection*)
 [PHP 3>= 3.0.7, PHP 4]

[ocicommit\(\)](#) valide toutes les transactions en cours sur la connexion Oracle *connection*.

10.48.9 ocirollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocirollback](#) (int *connection*)
 [PHP 3>= 3.0.7, PHP 4]

[ocirollback\(\)](#) annule les transactions en cours sur la connexion Oracle *connection*.

10.48.10 ocinewdescriptor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ocinewdescriptor](#) (int *connection*, int *type*)

[PHP 3>= 3.0.7, PHP 4]

[ocinewdescriptor\(\)](#) alloue l'espace nécessaire pour stocker un descripteur, ou un pointeur de LOB. Les valeurs acceptées pour type sont OCI_D_FILE, OCI_D_LOB et OCI_D_ROWID.

OCINewDescriptor

```
<?php
/* Ce script est fait pour être appelé dans un formulaire HTML
 * Il attends les variables $user, $password, $table, $where, et $commitsize
 * Le scrip efface alors les lignes sélectionnées avec ROWID et valide
 * l'effacement après chaque groupe de $commitsize lignes.
 * (Utilisez avec prudence, car il n'y a pas d'annulation possible).
 */
$conn = OCILogon($user, $password);
$stmt = OCIParse($conn,"select rowid from $table $where");
$rowid = OCINewDescriptor($conn,OCI_D_ROWID);
OCIDefineByName($stmt,"ROWID",&$rowid);
OCIExecute($stmt);
while ( OCIFetch($stmt) ) {
    $nrows = OCIRowCount($stmt);
    $delete = OCIParse($conn,"delete from $table where ROWID = :rid");
    OCIBindByName($delete,":rid",&$rowid,-1,OCI_B_ROWID);
    OCIExecute($delete);
    print "$nrows\n";
    if ( ($nrows % $commitsize) == 0 ) {
        OCICommit($conn);
    }
    $nrows = OCIRowCount($stmt);
    print "$nrows effacées...\n";
    OCIFreeStatement($stmt);
    OCILogoff($conn);
?>

<?php
/* Ce script est fait pour être appelé depuis un formulaire HTML.
 * Il attends les variables $user, $password, $table, $where, et $commitsize,
 * données par le formulaire. Le script efface
 * les lignes sélectionnées avec ROWID est valide les transactions
 * à chaque jeu de $commitsize lignes. (Attention : il n'y plus d'annulation) */
if(!isset($lob_upload) || $lob_upload == 'none'){
?>
<form action="upload.php3" method="post" enctype="multipart/form-data">
Upload file: <input type="file" name="lob_upload"><br>
<input type="submit" value="Upload"> - <input type="reset">
</form>
<?php
    } else {
        // $lob_upload contains the temporary filename of the uploaded file
        $conn = OCILogon($user, $password);
        $lob = OCINewDescriptor($conn, OCI_D_LOB);
        $stmt = OCIParse($conn,"insert into $table (id, the_blob) values(my_seq.NEXTVAL, EMPTY_BLOB)");
        OCIBindByName($stmt, ':the_blob', &$lob, -1, OCI_B_BLOB);
        OCIExecute($stmt);
        if($lob->savefile($lob_upload)){
            OCICommit($conn);
            echo "Blob sauvé!\n";
        }else{
            echo "Impossible de sauver le Blob\n";
        }
    }
}
```

```

    }
    OCIFreeDescriptor($lob);
    OCIFreeStatement($stmt);
    OCILogoff($conn);
  }
?>

```

10.48.11 ocirowcount

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocirowcount](#) (int *statement*)

[PHP 3>= 3.0.7, PHP 4]

[ocirowcount\(\)](#) retourne le nombre de lignes affectées par une commande de modification. Cette fonction ne vous indiquera pas le nombre de lignes retournées par un SELECT : il faut que les lignes aient été modifiées.

OCIRowCount

```

<?php
print "<HTML><PRE>";
    $conn = OCILogon("scott","tiger");
    $stmt = OCIParse($conn,"create table emp2 as select * from emp");
    OCIExecute($stmt);
    print OCIRowCount($stmt) . " rows inserted.<BR>";
    OCIFreeStatement($stmt);
    $stmt = OCIParse($conn,"delete from emp2");
    OCIExecute($stmt);
    print OCIRowCount($stmt) . " rows deleted.<BR>";
    OCICommit($conn);
    OCIFreeStatement($stmt);
    $stmt = OCIParse($conn,"drop table emp2");
    OCIExecute($stmt);
    OCIFreeStatement($stmt);
    OCILogOff($conn);
print "</PRE></HTML>";
?>

```

10.48.12 ocinumcols

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocinumcols](#) (int *stmt*)

[PHP 3>= 3.0.4, PHP 4]

[ocinumcols\(\)](#) retourne le nombre de colonnes dans un résultat.

OCINumCols

```

<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
    OCIExecute($stmt);
    while ( OCIFetch($stmt) ) {

```



```

print "\n";
    $ncols = OCINumCols($stmt);
    for ( $i = 1; $i <= $ncols; $i++ ) {
        $column_name = OCIColumnName($stmt,$i);
        $column_value = OCIResult($stmt,$i);
        print $column_name . ': ' . $column_value . "\n";
    }
print "\n";
}
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>

```

10.48.13 ocireresult

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [ocireresult](#)(int *statement*, mixed *column*)

[PHP 3>= 3.0.4, PHP 4]

[ocireresult\(\)](#) retourne les données de la colonne *column* dans la ligne courante (voir [ocifetch\(\)](#)).
[ocifetch\(\)](#) retournera tout les types, sauf les types abstraits (ROWIDs, LOBs et FILES).

10.48.14 ocifetch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifetch](#)(int *statement*)

[PHP 3>= 3.0.4, PHP 4]

[ocifetch\(\)](#) place la prochaine ligne (d'une commande SELECT) dans le pointeur interne de résultat.

10.48.15 ocifetchinto

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifetchinto](#)(int *stmt*, array *&result*, int *mode*)

[PHP 3>= 3.0.4, PHP 4]

[ocifetchinto\(\)](#) retourne la ligne suivante (pour une commande SELECT) dans le tableau *result*.

[ocifetchinto\(\)](#) écrasera le contenu de *result*. Par défaut, *result* sera un tableau à index numérique, commençant à 1, et qui contiendra toute les colonnes qui ne sont pas NULL.

L'option *mode* vous permet de modifier le comportement par défaut de la fonction. Vous pouvez passer plusieurs modes simplement en les additionnant (i.e. OCI_ASSOC+OCI_RETURN_NULLS). Les modes valides sont :

- OCI_ASSOC Retourne un tableau associatif.
- OCI_NUM Retourne un tableau à index numérique (DEFAULT, valeur par défaut)
- OCI_RETURN_NULLS Retourne les colonnes vides.
- OCI_RETURN_LOBS Retourne la valeur des objets LOB plutôt que leur descripteur.

10.48.16 ocifetchstatement

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifetchstatement](#) (int *stmt*, array &*variable*)

[PHP 3>= 3.0.8, PHP 4]

[ocifetchstatement\(\)](#) retourne toutes les lignes d'un résultat dans le tableau variable.

[ocifetchstatement\(\)](#) retourne le nombre de lignes retournées.

OCIFetchStatement

```

<?php
/* exemple OCIFetchStatement par mbritton@verinet.com (990624) */
$conn = OCILogon("scott","tiger");
$stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
$rows = OCIFetchStatement($stmt,$results);
if ( $rows > 0 ) {
print "<TABLE BORDER=\\"1\ ">\n";
print "<TR>\n";
while ( list( $key, $val ) = each( $results ) ) {
print "<TH>$key</TH>\n";
}
print "</TR>\n";
for ( $i = 0; $i < $rows; $i++ ) {
reset($results);
print "<TR>\n";
while ( $column = each($results) ) {
    $data = $column['value'];
print "<TD>$data[$i]</TD>\n";
}
print "</TR>\n";
}
print "</TABLE>\n";
} else {
echo "Rien n'a été trouvé<BR>\n";
}
print "$rows Records Selected<BR>\n";
OCIFreeStatement($stmt);
OCILogoff($conn);
?>

```

10.48.17 ocicolumnisnull

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocicolumnisnull](#) (int *stmt*, mixed *column*)

[PHP 3>= 3.0.4, PHP 4]

[ocicolumnisnull\(\)](#) retourne TRUE si la colonne *col* du résultat *stmt* est NULL. Vous pouvez utiliser le numéro de colonne (l'indexation des colonnes commence à 1) ou le nom de la colonne, pour le paramètre *col*.

10.48.18 ocicolumnname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ocicolumnname](#)(int *stmt*, int *col*)

[PHP 3>= 3.0.4, PHP 4]

[ocicolumnname\(\)](#) retourne le nom de la colonne numéro *col* (en commençant à 1).

OCIColumnName

```
<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
print "<TABLE BORDER=\"1\">";
print "<TR>";
print "<TH>Name</TH>";
print "<TH>Type</TH>";
print "<TH>Length</TH>";
print "</TR>";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_type = OCIColumnType($stmt,$i);
    $column_size = OCIColumnSize($stmt,$i);
print "<TR>";
print "<TD>$column_name</TD>";
print "<TD>$column_type</TD>";
print "<TD>$column_size</TD>";
print "</TR>";
    }
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>
```

Voir aussi [ocinumcols\(\)](#), [ocicolumntype\(\)](#), et [ocicolumnsize\(\)](#).

10.48.19 ocicolumnsize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocicolumnsize](#)(int *stmt*, mixed *column*)

[PHP 3>= 3.0.4, PHP 4]

[ocicolumnsize\(\)](#) retourne la taille de la colonne. Vous pouvez utiliser l'index de colonne (l'indexation commence à 1) ou le nom de la colonne dans le paramètre *col*.

OCIColumnSize

```
<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
print "<TABLE BORDER=\"1\">";
print "<TR>";
```

```

print "<TH>Name</TH>";
print "<TH>Type</TH>";
print "<TH>Length</TH>";
print "</TR>";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_type = OCIColumnType($stmt,$i);
    $column_size = OCIColumnSize($stmt,$i);
print "<TR>";
print "<TD>$column_name</TD>";
print "<TD>$column_type</TD>";
print "<TD>$column_size</TD>";
print "</TR>";
}
print "</TABLE>";
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>

```

Voir aussi [ocinumcols\(\)](#), [ocicolumnname\(\)](#) et [ocicolumnsize\(\)](#).

10.48.20 ocicolumntype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [ocicolumnname](#) (int *stmt*, int *col*)

[PHP 3>= 3.0.4, PHP 4]

[ocicolumntype\(\)](#) retourne le type de données de la colonne correspondant au numéro de colonne (les colonnes sont indexées à partir de 1).

OCIColumnType

```

<?php
print "<HTML><PRE>\n";
    $conn = OCILogon("scott", "tiger");
    $stmt = OCIParse($conn,"select * from emp");
OCIExecute($stmt);
print "<TABLE BORDER=\"1\">";
print "<TR>";
print "<TH>Name</TH>";
print "<TH>Type</TH>";
print "<TH>Length</TH>";
print "</TR>";
    $ncols = OCINumCols($stmt);
for ( $i = 1; $i <= $ncols; $i++ ) {
    $column_name = OCIColumnName($stmt,$i);
    $column_type = OCIColumnType($stmt,$i);
    $column_size = OCIColumnSize($stmt,$i);
print "<TR>";
print "<TD>$column_name</TD>";
print "<TD>$column_type</TD>";
print "<TD>$column_size</TD>";
print "</TR>";
}

```

```
OCIFreeStatement($stmt);
OCILogoff($conn);
print "</PRE>";
print "</HTML>\n";
?>
```

Voir aussi [ocinumcols\(\)](#), [ocicolumnname\(\)](#), et [ocicolumnsize\(\)](#).

10.48.21 ociserverversion

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ociserverversion](#) (int *conn*)
[PHP 3>= 3.0.4, PHP 4]

[ociserverversion\(\)](#) retourne une chaîne contenant les informations de version du serveur
OCIServerVersion

```
<?php
$conn = OCILogon("scott","tiger");
print "Version du serveur : " . OCIServerVersion($conn);
OCILogOff($conn);
?>
```

10.48.22 ocistatementtype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ocistatementtype](#) (int *stmt*)
[PHP 3>= 3.0.5, PHP 4]

[ocistatementtype\(\)](#) retourne une des valeurs suivantes :

1. "SELECT"
2. "UPDATE"
3. "DELETE"
4. "INSERT"
5. "CREATE"
6. "DROP"
7. "ALTER"
8. "BEGIN"
9. "DECLARE"
10. "UNKNOWN"

Exemples

```
<?php
print "<HTML><PRE>";
```

```

$conn = OCILogon("scott","tiger");
$sql = "delete from emp where deptno = 10";
$stmt = OCIParse($conn,$sql);
if ( OCISTatementType($stmt) == "DELETE" ) {
die "Vous n'etes pas autorisé à effacer dans cette table.<BR>";
}
OCILogoff($conn);
print "</PRE></HTML>";
?>

```

10.48.23 ocinewcursor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocinewcursor](#) (int *conn*)

[PHP 3>= 3.0.8, PHP 4]

[ocinewcursor\(\)](#) alloue un nouveau pointeur de commande, pour la connexion *conn*.

Utiliser un REF CURSOR issue d'une procédure enregistrée.

```

<?php
// supposons que votre procédure stockée info.output retourne un pointeur
// de curseur dans : data
$conn = OCILogon("scott","tiger");
$curs = OCINewCursor($conn);
$stmt = OCIParse($conn,"begin info.output(:data); end;");
ocibindbyname($stmt,"data",&$curs,-1,OCI_B_CURSOR);
ociexecute($stmt);
ociexecute($curs);
while (OCIFetchInto($curs,&$data)) {
var_dump($data);
}
OCIFreeCursor($stmt);
OCIFreeStatement($curs);
OCILogoff($conn);
?>

```

Utiliser un REF CURSOR issue d'une commande SELECT

```

<?php
print "<HTML><BODY>";
$conn = OCILogon("scott","tiger");
$count_cursor = "CURSOR(select count(empno) num_ems from emp " .
                "where emp.deptno = dept.deptno) as EMPCNT from dept";
$stmt = OCIParse($conn,"select deptno,dname,$count_cursor");
ociexecute($stmt);
print "<TABLE BORDER=\"1\">";
print "<TR>";
print "<TH>DEPT NAME</TH>";
print "<TH>DEPT #</TH>";
print "<TH># EMPLOYEES</TH>";
print "</TR>";
while (OCIFetchInto($stmt,&$data,OCI_ASSOC)) {

```

```

print "<TR>";
    $dname = $data["DNAME"];
    $deptno = $data["DEPTNO"];
print "<TD>$dname</TD>";
print "<TD>$deptno</TD>";
ociexecute($data["EMPCNT"]);
while (OCIFetchInto($data["EMPCNT"],&$subdata,OCI_ASSOC)) {
    $num_emps = $subdata["NUM_EMPS"];
print "<TD>$num_emps</TD>";
}
print "</TR>";
}
print "</TABLE>";
print "</BODY></HTML>";
OCIFreeStatement($stmt);
OCILogoff($conn);
?>

```

10.48.24 ocifreestatement

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifreestatement](#) (int *stmt*)
 [PHP 3>= 3.0.5, PHP 4]

[ocifreestatement\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

10.48.25 ocifreecursor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifreecursor](#) (int *stmt*)
 [PHP 3>= 3.0.8, PHP 4]

[ocifreecursor\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

10.48.26 ocifreedesc

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ocifreedesc](#) (object *lob*)
 [PHP 4 >= 4.0b4]

[ocifreedesc\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'échec.

10.48.27 ociparse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ociparse](#) (int *conn*, string *query*)
 [PHP 3>= 3.0.4, PHP 4]

[ociparse\(\)](#) analyse la requête *query* sur la connexion *conn*, et retourne TRUE si la requête *query* est valide, et

FALSE, si ce n'est pas le cas. *query* peut être n'importe quelle requête SQL.

10.48.28 ocierror

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [ocierror](#) (int *stmt*)*conn*
[PHP 3>= 3.0.7, PHP 4]

[ocierror\(\)](#) retourne la dernière erreur trouvée. Si l'option *stmt/conn* n'est pas fournie, la dernière erreur rencontrée est retournée. Si aucune erreur n'est trouvée, [ocierror\(\)](#) retourne FALSE.

10.48.29 ociinternaldebug

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ociinternaldebug](#) (int *onoff*)
[PHP 3>= 3.0.4, PHP 4]

[ociinternaldebug\(\)](#) active l'affichage des informations de debuggage. Pour les afficher, mettez onoff à 0, ou sinon à 1 pour les cacher.

10.49 OpenSSL

[\[Notes en ligne\]](#)

Cette extension utilise les fonctions de [OpenSSL](#) pour générer et vérifier les signatures, ainsi que pour sceller (chiffrer) et ouvrir (déchiffrer) les données. Vous avez besoin de OpenSSL >= 0.9.6 pour utiliser ce module. OpenSSL offre de nombreuses fonctionnalités qui ne sont pas encore supportées par ce module. Elle seront ajoutées ultérieurement.

10.49.1 openssl_free_key

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [openssl_free_key](#) (int *key_identifieur*)

[openssl_free_key\(\)](#) libère les ressources associées à *key_identifieur*.

10.49.2 openssl_get_privatekey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [openssl_get_privatekey](#) (string *key*, string *passphrase*)

[openssl_get_privatekey\(\)](#) retourne un identifiant de clé positif, ou FALSE en cas d'erreur.

[openssl_get_privatekey\(\)](#) analyse la clé privée *key*, au format PEM, et la prépare pour à être utilisée par d'autres fonctions. Le paramètre optionnel *passphrase* doit être utilisé si la clé est chiffrée (protégée par un mot de passe).

10.49.3 openssl_get_publickey

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [openssl_get_publickey](#)(string *certificate*)

[openssl_get_publickey\(\)](#) retourne un identifiant de clé positif, ou FALSE en cas d'erreur.

[openssl_get_publickey\(\)](#) extrait la clé publique du certificat *certificate* (format X.509), et la prépare à être utilisée ultérieurement.

10.49.4 openssl_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [openssl_open](#)(string *sealed_data*, string *open_data*, string *env_key*, int *priv_key_id*)

[openssl_open\(\)](#) TRUE en cas de succès, et FALSE sinon. En cas de succès, les données déchiffrées sont placées dans *open_data*.

[openssl_open\(\)](#) ouvre (déchiffre) les données *sealed_data* en utilisant la clé privée *priv_key_id* et la clé d'enveloppe *env_key*. La clé d'enveloppe est générée lorsque les données sont scellées, et ne peut être utilisée qu'avec la clé privée spécifique. Reportez vous à [openssl_seal\(\)](#) pour plus d'informations.

Exemple avec openssl_open()

```

<?php
// On suppose que $sealed et $env_key contiennent les données scellées
// et la clé d'enveloppe, fournies par l'expéditeur
// lecture de la clé privée dans un fichier
$fp = fopen("/src/openssl-0.9.6/demos/sign/key.pem", "r");
$priv_key = fread($fp, 8192);
fclose($fp);
$pkeyid = openssl_get_privatekey($priv_key);
// déchiffrement des données : elles sont placées dans $open
if (openssl_open($sealed, $open, $env_key, $pkeyid))
echo "Voici les données déchiffrées : ", $open;
else
echo "Impossible de déchiffrer les données";
// libération des ressources
openssl_free_key($pkeyid);
?>

```

Voir aussi [openssl_seal\(\)](#).

10.49.5 openssl_seal

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [openssl_seal](#)(string *data*, string *sealed_data*, array *env_keys*, array *pub_key_ids*)

[openssl_seal\(\)](#) retourne la longueur des données scellées en cas de succès, et FALSE sinon. En cas de succès, les données scellées sont placées dans le paramètre *sealed_data*, et les clés d'enveloppe dans *env_keys*.

[openssl_seal\(\)](#) scelle (chiffre) les données *data* en utilisant l'algorithme RC4 avec une clé secrète générée

aléatoirement. La clé est chiffrée avec chaque clé publique associée à *pub_key_ids* et chaque clé ainsi encryptée est retournée dans *env_keys*. Cela signifie que vous pouvez envoyer des données scellées à plusieurs destinataires (en supposant que chacun ait reçu la clé publique). Chaque destinataire doit recevoir les données encryptées et la clé d'enveloppe, qui a été encryptée avec la clé publique du destinataire.

Exemple avec openssl_seal()

```
<?php
// On suppose que $data contient les données à sceller
// lecture de la clé publique pour chaque destinataire
$fp = fopen("/src/openssl-0.9.6/demos/maurice/cert.pem", "r");
$cert = fread($fp, 8192);
fclose($fp);
$pk1 = openssl_get_publickey($cert);
// pour le deuxième destinataire
$fp = fopen("/src/openssl-0.9.6/demos/sign/cert.pem", "r");
$cert = fread($fp, 8192);
fclose($fp);
$pk2 = openssl_get_publickey($cert);
// scelle le message : seuls, les possesseurs de $pk1 et $pk2 peuvent déchiffrer
// le message $sealed avec les clés $keys[0] et $keys[1] (respectivement).
openssl_seal($data, $sealed, $keys, array($pk1,$pk2));
// libère les clés de la mémoire
openssl_free_key($pk1);
openssl_free_key($pk2);
?>
```

Voir aussi [openssl_open\(\)](#).

10.49.6 openssl_sign

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [openssl_sign](#) (string *data*, string *signature*, int *priv_key_id*)

[openssl_sign\(\)](#) retourne TRUE en cas de succès, et FALSE sinon. En cas de succès, la signature est placée dans *signature*.

[openssl_sign\(\)](#) calcule la signature des données *data* en utilisant l'algorithme SHA1 (hashing) suivi du chiffage avec la clé privée *priv_key_id*. Notez que les données elles-mêmes ne sont pas chiffrées.

Exemple avec openssl_sign()

```
<?php
// On suppose que $data contient les données à signer
// lecture de la clé publique pour chaque destinataire
$fp = fopen("/src/openssl-0.9.6/demos/sign/key.pem", "r");
$priv_key = fread($fp, 8192);
fclose($fp);
$pkeyid = openssl_get_privatekey($priv_key);
// calcule de la signature
openssl_sign($data, $signature, $pkeyid);
// libère les clés de la mémoire
openssl_free_key($pkeyid);
?>
```

Voir aussi [openssl_verify\(\)](#).

[10.49.7 openssl_verify](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [openssl_verify](#) (string *data*, string *signature*, int *pub_key_id*)

[openssl_verify\(\)](#) retourne 1 si la signature est correcte, 0 si la signature est incorrecte, et -1 en cas d'erreur. [openssl_verify\(\)](#) vérifie que la signature *signature* est correcte pour les données *data*, et avec la clé publique *pub_key_id*. Cette clé doit être la clé publique correspondant à la clé privée utilisée lors de la signature.

Exemple avec openssl_verify()

```
<?php
// On suppose que $data et $signature contiennent les données à signer et
// la signature
// lecture de la clé publique depuis le certificat
$fp = fopen("/src/openssl-0.9.6/demos/sign/cert.pem", "r");
$cert = fread($fp, 8192);
fclose($fp);
$pubkeyid = openssl_get_publickey($cert);
// indique si la signature est correcte
$ok = openssl_verify($data, $signature, $pubkeyid);
if ($ok == 1)
echo "Signature valide";
elseif ($ok == 0)
echo "Signature erronée";
else
echo "Erreur de vérification de la signature";
// libère les clés de la mémoire
openssl_free_key($pubkeyid);
?>
```

Voir aussi [openssl_sign\(\)](#).

[10.50 Oracle](#)

[\[Notes en ligne\]](#)

[10.50.1 ora_bind](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_bind](#) (int *cursor*, string *PHP variable name*, string *SQL parameter name*, int *length*, int *type*)
[PHP 3, PHP 4]

[ora_bind\(\)](#) retourne TRUE si la liaison a pu se faire, et sinon FALSE. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Cette fonction lie une variable PHP avec un paramètre SQL. Le paramètre SQL doit être de la forme ":name". Avec l'option, vous pouvez choisir si le paramètre SQL est de type entrée/sortie (0, valeur par défaut), entrée seulement (1) ou sortie seulement (2). Comme dans PHP 3.0.1, vous pouvez respectivement utiliser les

constantes `ORA_BIND_INOUT`, `ORA_BIND_IN` et `ORA_BIND_OUT` plutôt que des nombres. [ora_bind\(\)](#) doit être appelée après la fonction [ora_parse\(\)](#) et avant [ora_exec\(\)](#). Les valeurs d'entrées peuvent alors être fournies par assignation des variables PHP. Après la fonction [ora_exec\(\)](#) les variables liées contiennent les valeurs de sortie, si elles sont disponibles. Par exemple :

```
<?php
ora_parse($curs, "declare tmp INTEGER; begin tmp := :in; :out := tmp; :x := 7.77; end;");
ora_bind($curs, "result", ":x", $len, 2);
ora_bind($curs, "input", ":in", 5, 1);
ora_bind($curs, "output", ":out", 5, 2);
$input = 765;
ora_exec($curs);
echo "Résultat: $result<BR>Sortie: $output<BR>Entrée: $input";
?>
```

10.50.2 ora_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_close](#)(int *cursor*)

[PHP 3, PHP 4]

[ora_close\(\)](#) retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

[ora_close\(\)](#) termine les pointeurs ouverts avec la fonction [ora_open\(\)](#).

10.50.3 ora_columnname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ora_columnname](#)(int *cursor*, int *column*)

[PHP 3, PHP 4]

[ora_columnname\(\)](#) retourne le nom du champs *column* du pointeur *cursor*. Le nom retourné sera en majuscule.

10.50.4 ora_columnsize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_columnsize](#)(int *cursor*, int *column*)

[PHP 3, PHP 4]

[ora_columnsize\(\)](#) retourne la taille de la colonne *column* dans le résultat *cursor*.

10.50.5 ora_columntype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ora_columntype](#)(int *cursor*, int *column*)

[PHP 3, PHP 4]

[ora_columntype\(\)](#) retourne le type de la colonne *column* du résultat *cursor*. Le type retourné prendra une des valeurs suivantes :

- "VARCHAR2"
- "VARCHAR"
- "CHAR"
- "NUMBER"
- "LONG"
- "LONG RAW"
- "ROWID"
- "DATE"
- "CURSOR"

10.50.6 ora_commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_commit](#) (int *conn*)
[PHP 3, PHP 4]

[ora_commit\(\)](#) retourne TRUE si la validation a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

[ora_commit\(\)](#) valide les transactions Oracle. Une transaction est définie par toutes les requêtes effectuées sur la connexion conn depuis la dernière validation ou annulation (avec auto-validation inactivée) ou depuis l'établissement de la connexion.

10.50.7 ora_commitoff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_commitoff](#) (int *conn*)
[PHP 3, PHP 4]

[ora_commitoff\(\)](#) retourne TRUE si la désactivation a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

[ora_commitoff\(\)](#) inactive la validation automatique après chaque [ora_exec\(\)](#).

10.50.8 ora_commiton

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_commiton](#) (int *conn*)
[PHP 3, PHP 4]

[ora_commiton\(\)](#) active la validation automatique après chaque [ora_exec\(\)](#).

[ora_commiton\(\)](#) retourne TRUE si l'activation a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.50.9 ora_do

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_do](#)(int *conn*, string *query*)

[PHP 3, PHP 4]

[ora_do\(\)](#) est une combinaison de [ora_parse\(\)](#), [ora_exec\(\)](#) et [ora_fetch\(\)](#). Elle va analyser la requête, l'exécuter et lire la première ligne du résultat.

[ora_do\(\)](#) retourne TRUE en cas de succès, et FALSE sinon. Les détails sur les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Voir aussi [ora_parse\(\)](#), [ora_exec\(\)](#), et [ora_fetch\(\)](#).

10.50.10 ora_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ora_error](#)(int *cursor_or_connection*)

[PHP 3, PHP 4]

[ora_error\(\)](#) retourne un message d'erreur de la forme XXX-NNNNN avec XXX qui est l'origine de l'erreur, et NNNNN qui identifie le message d'erreur. Note : *Le support des connexions a été ajouté dans PHP 3.0.4.*

Avec les versions UNIX d'Oracle, vous pouvez avoir des messages d'erreur tels que : \$ oerr ora 00001 00001, 00000, "unique constraint (%s.%s) violated" // *Cause: An update or insert statement attempted to insert a duplicate key // For Trusted ORACLE configured in DBMS MAC mode, you may see // this message if a duplicate entry exists at a different level. // *Action: Either remove the unique restriction or do not insert the key .

10.50.11 ora_errorcode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_errorcode](#)(int *cursor_or_connection*)

[PHP 3, PHP 4]

[ora_errorcode\(\)](#) retourne le code d'erreur numérique de la dernière commande exécutée sur la connexion ou le pointeur fourni en paramètre. Note : *Les identifiants de connexion ne sont acceptés qu'à partir de la version 3.0.4.*

10.50.12 ora_exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_exec](#)(int *cursor*)

[PHP 3, PHP 4]

[ora_exec\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'erreur. L'erreur générée sera alors accessible avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.50.13 ora_fetch

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_fetch](#)(int *cursor*)

[PHP 3, PHP 4]

[ora_fetch\(\)](#) retourne TRUE (une ligne a été lue) ou FALSE (plus de lignes à lire ou erreur). Si une erreur survient, sa valeur sera disponible dans les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Lit une ligne de données sur le pointeur cursor.

Voir aussi [ora_parse\(\)](#), [ora_exec\(\)](#), et [ora_do\(\)](#).

10.50.14 ora_fetch_into

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_fetch_into](#)(int *cursor*, array *result*, int *flags*)

[PHP 3, PHP 4]

[ora_fetch_into\(\)](#) lit la ligne courante du résultat *cursor* dans le tableau *result*.

Exemple ora_fetch_into()

```
<?php
array($results);
ora_fetch_into($cursor, &$results);
echo $results[0];
echo $results[1];
?>
```

Notez que vous devez passer le tableau par référence;

Voir aussi [ora_parse\(\)](#), [ora_exec\(\)](#), [ora_fetch\(\)](#), et [ora_do\(\)](#).

10.50.15 ora_getcolumn

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [ora_getcolumn](#)(int *cursor*, mixed *column*)

[PHP 3, PHP 4]

[ora_getcolumn\(\)](#) retourne la valeur de la colonne. Si une erreur survient, FALSE est retourné et [ora_errorcode\(\)](#) aura une valeur non nulle. Notez, qu'un test à FALSE, avec cette fonction peut être TRUE, même sans erreur : en effet, la fonction peut retourner des valeurs telles que (résultat NULL, chaînes vides, nombre 0, la chaîne "0").

10.50.16 ora_logoff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_logoff](#)(int *connection*)

[PHP 3, PHP 4]

[ora_logoff\(\)](#) retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

Déconnecte l'utilisateur, et se déconnecte.

Voir aussi [ora_logon\(\)](#).

[10.50.17 ora_logon](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_logon](#) (string *user*, string *password*)

[PHP 3, PHP 4]

[ora_logon\(\)](#) établit une connexion entre PHP et un serveur Oracle avec les noms d'utilisateur *user* et le mot de passe *password*.

Les connexions peut être faites avec *SQL*Net* en fournissant le nom *TNS* de la manière suivante :

```
<?php
$conn = ora_logon("user@TNSNAME", "pass");
?>
```

Si vous avez des données qui ne sont pas ASCII, vous devriez vérifier que la variable *NLS_LANG* a été correctement configuré dans votre environnement. Pour les modules de serveur, vous devrez la configurer dans l'environnement d'exécution du serveur avant de le lancer.

[ora_logon\(\)](#) retourne un index de connexion, en cas de succès, ou FALSE en cas d'échec. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

[10.50.18 ora_plogon](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_plogon](#) (string *user*, string *password*)

[PHP 3, PHP 4]

[ora_plogon\(\)](#) établit une connexion persistante à un serveur Oracle, avec l'utilisateur *user* et le mot de passe *password*.

Voir aussi [ora_logon\(\)](#).

[10.50.19 ora_numcols](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_numcols](#) (int *cursor_ind*)

[PHP 3, PHP 4]

[ora_numcols\(\)](#) retourne le nombre de colonne dans le résultat *cursor_ind*. Cette valeur n'est significative qu'après une requête parse/exec/fetch.

Voir aussi [ora_parse\(\)](#), [ora_exec\(\)](#), [ora_fetch\(\)](#), et [ora_do\(\)](#).

[10.50.20 ora_numrows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ora_numrows](#) (int *cursor_ind*)

[PHP 3, PHP 4]

[ora_numrows\(\)](#) retourne le nombre de colonnes dans le résultat *cursor_ind*.

10.50.21 ora_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)int [ora_open](#)(int *connection*)

[PHP 3, PHP 4]

[ora_open\(\)](#) ouvre un pointeur Oracle sur la connexion.Retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.50.22 ora_parse

[\[Notes en ligne\]](#) [\[Exemples\]](#)int [ora_parse](#)(int *cursor_ind*, string *sql_statement*, int *defer*)

[PHP 3, PHP 4]

[ora_parse\(\)](#) analyse une requête SQL ou un bloc PL/SQL et l'associe avec le pointeur *cursor_ind*. Retourne 0 en cas de succès, et -1 en cas d'erreur.

Retourne 0 en cas de succès, et -1 en cas d'erreur.

Voir aussi [ora_exec\(\)](#), [ora_fetch\(\)](#), et [ora_do\(\)](#).

10.50.23 ora_rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)int [ora_rollback](#)(int *connection*)

[PHP 3, PHP 4]

[ora_rollback\(\)](#) annule une transaction Oracle. (Voir [ora_commit\(\)](#) pour la définition d'une transaction).Retourne TRUE si la fermeture a bien eu lieu, et FALSE sinon. Les erreurs sont accessibles avec les fonctions [ora_error\(\)](#) et [ora_errorcode\(\)](#).

10.51 Entrées/sorties

[\[Notes en ligne\]](#)

Les fonctions d'entrée/sorties vous permettent de contrôler quand les données sont envoyées par le script. Cela peut être utile dans certaines situations, notamment si vous devez envoyer des entêtes au navigateur après avoir envoyé des données. Ces fonctions n'affectent pas les entêtes envoyés par la fonction [header\(\)](#) ou les cookies envoyés par [setcookie\(\)](#). Seules les fonctions telles que [echo\(\)](#) et les données entre blocs PHP sont affectés.

Exemple de gestion des sorties

```
<?php
ob_start();
```

```

echo "Bonjour\n";
setcookie ("nom_du_cookie", "valeur_du_cookie");
ob_end_flush();
?>

```

Dans l'exemple ci-dessus, la fonction [echo\(\)](#) est stockée dans un buffer jusqu'à l'appel de la fonction [ob_end_flush\(\)](#) was called. Dans le même temps, l'appel à [setcookie\(\)](#) a réussi à créer un cookie, sans générer d'erreur. (D'habitude, vous devez envoyer les entêtes avant les données). Voir aussi [header\(\)](#) et [setcookie\(\)](#).

10.51.1 flush

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [flush](#)

[PHP 3, PHP 4]

Vide les buffers de sortie de PHP et tous ceux que PHP utilisait (CGI, un serveur web, etc.).

Note : [flush\(\)](#) n'a aucun effet sur la bufferisation de votre serveur web ou du navigateur.

De nombreux serveur, essentiellement sous Windows, continueront à bufferiser l'affichage de votre script jusqu'à ce qu'il soit terminé, avant de transmettre les résultats à l'internaute.

Même le navigateur peut mettre des informations en cache avant de les afficher. Par exemple, Netscape écrit les textes dans un cache, jusqu'à ce qu'il ai reçu une fin de ligne, ou une balise ouvrante. Il n'affichera pas les tables avant d'avoir reçu la balise fermante </table>.

10.51.2 ob_start

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_start](#) (string *output_callback*)

[PHP 4]

[ob_start\(\)](#) démarre la bufferisation de sortie. Tant qu'elle est enclenchée, aucune données n'est envoyée au client web, mais temporairement mis en buffer.

Le contenu de ce buffer peut être copié dans une chaîne avec la fonction [ob_get_contents\(\)](#). Pour afficher le contenu de ce buffer, utilisez [ob_end_flush\(\)](#). Au contraire, [ob_end_clean\(\)](#) effacera le contenu de ce buffer. Une fonction optionnelle de callback peut être spécifiée en troisième argument. [ob_start\(\)](#) prend une chaîne comme paramètre, et retourne une chaîne. Elle sera appelée par [ob_end_flush\(\)](#) et recevra le contenu du buffer de sortie. Elle doit retourner un nouveau contenu pour le buffer de sortie : celui ci sera envoyé à la sortie standard.

Les buffers de sortie sont gérés par pile, c'est à dire que vous pouvez appeler plusieurs

[ob_start\(\)](#) simultanément. Assurez-vous que vous appelez [ob_end_flush\(\)](#) suffisamment souvent. Si plusieurs fonctions de callback sont actives, les contenus seront filtrés séquentiellement, dans l'ordre d'emboîtement.

Exemple de callback

```

<?php
function c($str) {
    // Aa claar da la lana, man ama Paarrat..
    return nl2br(ereg_replace("[aeiou]", "u", $str));
}
function d($str) {

```

```

return strip_tags($str);
}
?>
<?php ob_start("c"); ?>
Au clair de la lune, mon ami Pierrot
<?php ob_start("d"); ?>
<h1>..Prete moi ta plume, pour ecrire un mot...</h1>
<?php ob_end_flush(); ?>
... Ma chandelle est morte, je n'ai plus de feu
<?php ob_end_flush(); ?>
... Ouvre moi la porte, pour l'amour de Dieu
<?php ob_end_flush(); ?>

```

Voir aussi [ob_get_contents\(\)](#), [ob_end_flush\(\)](#), [ob_end_clean\(\)](#), et [ob_implicit_flush\(\)](#).

10.51.3 ob_get_contents

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ob_get_contents](#)

[PHP 4]

[ob_get_contents\(\)](#) retourne le contenu du buffer de sortie si la bufferisation est active, ou FALSE sinon.

Voir aussi [ob_start\(\)](#) et [ob_get_length\(\)](#).

10.51.4 ob_get_length

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ob_get_length](#)

[PHP 4 >= 4.0.2]

[ob_get_length\(\)](#) retourne la longueur du contenu du buffer de sortie si la bufferisation est activée, et FALSE sinon.

Voir aussi [ob_start\(\)](#) et [ob_get_contents\(\)](#).

10.51.5 ob_end_flush

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_end_flush](#)

[PHP 4]

[ob_end_flush\(\)](#) envoie le contenu du buffer de sortie (s'il existe) et éteint la bufferisation de sortie. Si vous voulez continuer à manipuler la valeur du buffer, vous pouvez appeler [ob_get_contents\(\)](#) avant [ob_end_flush\(\)](#) car le contenu du buffer est détruit après un appel à [ob_end_flush\(\)](#).

Voir aussi [ob_start\(\)](#), [ob_get_contents\(\)](#), et [ob_end_clean\(\)](#).

10.51.6 ob_end_clean

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_end_clean](#)

[PHP 4]

[ob_end_clean\(\)](#) détruit les données du buffer de sortie, et éteint la bufferisation.
Voir aussi [ob_start\(\)](#) et [ob_end_flush\(\)](#).

10.51.7 ob_implicit_flush

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ob_implicit_flush](#) (int *flag*)
[PHP 4 >= 4.0b4]

[ob_implicit_flush\(\)](#) active/désactive l'envoi implicite (si *flag* est fourni. Par défaut, il est activé). L'envoi implicite signifie que toute fonction qui envoie des données au client web veront leurs données envoyées immédiatement (la fonction [flush\(\)](#) est appelée automatiquement).

Une fois que l'envoi implicite est désactivé, le buffer de sortie ne sera envoyé qu'au moment de l'appel de [ob_end_flush\(\)](#).

Voir aussi [flush\(\)](#), [ob_start\(\)](#) et [ob_end_flush\(\)](#).

10.52 Ovrimos SQL

[\[Notes en ligne\]](#)

Ovrimos SQL Server est une base de données relationnelle client/serveur et transactionnelle, combinée avec des fonctionnalités web, et des transactions rapides.

Ovrimos SQL Server est disponible à www.ovrimos.com. Pour activer le support ovrimos de PHP, il suffit de compiler PHP avec l'option '--with-ovrimos' du script de configuration. Vous devrez aussi installer la librairie sqlcli disponible avec la distribution Ovrimos SQL Server.

Connection au serveur Ovrimos SQL Server et selectionn d'une table système

```
<?php
$conn = ovrimos_connect ("server.domain.com", "8001", "admin", "password");
if ($conn != 0) {
    echo ("Connection établie!");
    $res = ovrimos_exec ($conn, "select table_id, table_name from sys.tables");
    if ($res != 0) {
        echo "Requête effectuée!";
        ovrimos_result_all ($res);
        ovrimos_free_result ($res);
    }
    ovrimos_close($conn);
}
?>
```

Cet exemple effectue une connexion réussie.

10.52.1 ovrimos_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_connect](#) (string *host*, string *db*, string *user*, string *password*)
[PHP 4 >= 4.0.3]

[ovrimos_connect\(\)](#) sert à se connecter à un serveur Ovrimos.

[ovrimos_connect\(\)](#) retourne un identifiant de connexion, supérieur à 0, ou 0 en cas d'échec. *host* est l'adresse

IP de l'hôte Ovrimos, et **db** est soit le nom d'une base de données, soit une chaîne contenant le numéro de port.

Exemple avec ovrimos_connect()

```
<?php
$conn = ovrimos_connect ("server.domain.com", "8001", "admin", "password");
if ($conn != 0) {
    echo "Connection établie!";
    $res=ovrimos_exec ($conn, "select table_id, table_name from sys.tables");
    if ($res != 0) {
        echo "Requête effectuée!";
        ovrimos_result_all ($res);
        ovrimos_free_result ($res);
    }
    ovrimos_close ($conn);
}
?>
```

L'exemple ci dessus montre comment se connecter à une base de donnée et afficher le contenu d'une table.

10.52.2 ovrimos_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ovrimos_close](#) (int *connection*)

[PHP 4 >= 4.0.3]

[ovrimos_close\(\)](#) sert à fermer une connexion à un serveur Ovrimos.

[ovrimos_close\(\)](#) ferme la connexion au serveur Ovrimos. Toutes les transactions non validées sont annulées.

10.52.3 ovrimos_close_all

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [ovrimos_close_all](#) (void)

[PHP 4 >= 4.0.3]

[ovrimos_close_all\(\)](#) sert à fermer toutes les connexions.

[ovrimos_close_all\(\)](#) ferme toutes les connexions à Ovrimos. Toutes les transactions non validées sont annulées.

10.52.4 ovrimos_longreadlen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_longreadlen](#) (int *result_id*, int *length*)

[PHP 4 >= 4.0.3]

[ovrimos_longreadlen\(\)](#) sert à lire la taille des données qui sera lues lors de l'accès à une colonne de grande taille.

[ovrimos_longreadlen\(\)](#) indique le nombre d'octets qui seront lus dans une colonne de grande taille (long varchar et long varbinary). Par défaut, 0. Indépendamment du fait que [ovrimos_longreadlen\(\)](#) requiert *result_id*, actuellement [ovrimos_longreadlen\(\)](#) affecte ce paramètre pour tous les résultats. Retourne vrai.

10.52.5 ovrimos_prepare

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_prepare](#)(int *connection_id*, string *query*)

[PHP 4 >= 4.0.3]

[ovrimos_prepare\(\)](#) sert à préparer une requête SQL.

[ovrimos_prepare\(\)](#) prépare une requête SQL et retourne un identifiant de résultat *result_id* (ou FALSE en cas d'échec).

Connexion à un serveur Ovrimos SQL Server et préparation d'une requête

```

<?php
$conn=ovrimos_connect ("db_host", "8001", "admin", "password");
if ($conn!=0) {
    echo "Connection établie!";
    $res=ovrimos_prepare ($conn, "select table_id, table_name
from sys.tables where table_id=1");
    if ($res != 0) {
        echo "Préparation faite!";
        if (ovrimos_execute ($res)) {
            echo "Exécution réussie!\n";
            ovrmos_result_all ($res);
        } else {
            echo "Exécution manquée!";
        }
        ovrmos_free_result ($res);
    } else {
        echo "Préparation manquée!\n";
    }
    ovrmos_close ($conn);
}
?>

```

Cet exemple montre comment se connecter à un serveur Ovrimos SQL Server, comment préparer une requête SQL et l'exécuter.

10.52.6 ovrimos_execute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_execute](#)(int *result_id*, array *parameters_array*)

[PHP 4 >= 4.0.3]

[ovrimos_execute\(\)](#) sert à exécuter une requête SQL.

[ovrimos_execute\(\)](#) exécute une requête préparée. Retourne TRUE ou FALSE. Si la requête préparée contient des paramètres (des points d'interrogations dans la requête), un nombre correct de paramètre doit être passé dans le tableau *parameters_array*. Notez que [ovrimos_execute\(\)](#) ne suit pas les conventions PHP qui placent les noms des paramètres entre crochets. L'auteur n'a pas pu s'y faire.

10.52.7 ovrimos_cursor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_cursor](#) (int *result_id*)

[PHP 4 >= 4.0.3]

[ovrimos_cursor\(\)](#) sert à lire le nom du curseur

[ovrimos_cursor\(\)](#) retourne le nom du curseur. Pratique, lorsqu'on veut faire des modifications ou des effacements avec des curseurs déjà positionnés.

10.52.8 [ovrimos_exec](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_exec](#) (int *connection_id*, string *query*)

[PHP 4 >= 4.0.3]

[ovrimos_exec\(\)](#) sert à exécuter une requête SQL.

[ovrimos_exec\(\)](#) exécute une requête SQL (selection ou modification), et retourne un identifiant de résultat *result_id* (ou bien FALSE, en cas d'échec). Evidemment, la requête SQL ne doit pas contenir de paramètres.

10.52.9 [ovrimos_fetch_into](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_fetch_into](#) (int *result_id*, array *result_array*, string *how* , int *rownumber*)

[PHP 4 >= 4.0.3]

[ovrimos_fetch_into\(\)](#) lit une ligne dans un résultat SQL.

[ovrimos_fetch_into\(\)](#) lit une ligne dans le résultat *result_id*, qui doit être passé en référence. La ligne qui sera lue est déterminée par les deux paramètres *how* et *rownumber*. *how* peut prendre les valeurs de 'Next' (suivant, valeur par défaut), 'Prev' (précédent), 'First' (premier), 'Last' (dernier), 'Absolute' (position absolue). La casse de *how* n'est pas prise en compte. *rownumber* est optionne, sauf dans le cas d'Absolute'. Retourne TRUE ou FALSE.

Lit un exemple

```
<?php
$conn=ovrimos_connect ("neptune", "8001", "admin", "password");
if ($conn!=0) {
    echo "Connection établie!";
    $res=ovrimos_exec ($conn,"SELECT table_id, table_name FROM sys.tables");
    if ($res != 0) {
        echo "Requête effectuée!";
        if (ovrimos_fetch_into ($res, &$row)) {
            list ($table_id, $table_name) = $row;
            echo "table_id=".$table_id.", table_name=".$table_name."\n";
            if (ovrimos_fetch_into ($res, &$row)) {
                list ($table_id, $table_name) = $row;
                echo "table_id=".$table_id.", table_name=".$table_name."\n";
            } else {
                echo "Next: erreur\n";
            }
        } else {
            echo "First: erreur\n";
        }
    }
}
```

```

ovrimos_free_result ($res);
    }
ovrimos_close ($conn);
}
?>

```

Cet exemple lis une ligne.

10.52.10 ovrinos fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_fetch_row](#) (int *result_id*, int *how* , int *row_number*)

[PHP 4 >= 4.0.3]

[ovrimos_fetch_row\(\)](#) lit une ligne dans un résultat SQL.

[ovrimos_fetch_row\(\)](#) lit une ligne dans un résultat. Les colonnes doivent être lues par un autre appel.

Retourne TRUE ou FALSE.

Exemple de lecture de ligne

```

<?php
$conn = ovrinos_connect ("remote.host", "8001", "admin", "password");
if ($conn != 0) {
    echo "Connection établie!";
    $res=ovrimos_exec ($conn, "select table_id, table_name from sys.tables");
    if ($res != 0) {
        echo "Requête effectuée!";
        if (ovrimos_fetch_row ($res, "First")) {
            $table_id = ovrinos_result ($res, 1);
            $table_name = ovrinos_result ($res, 2);
            echo "table_id=".$table_id.", table_name=".$table_name."\n";
            if (ovrimos_fetch_row ($res, "Next")) {
                $table_id = ovrinos_result ($res, "table_id");
                $table_name = ovrinos_result ($res, "table_name");
            } else {
                echo "table_id=".$table_id.", table_name=".$table_name."\n";
            }
        } else {
            echo "Next: erreur\n";
        }
    } else {
        echo "First: erreur\n";
    }
    ovrinos_free_result ($res);
}
ovrimos_close ($conn);
}
?>

```

Cet exemple lit une ligne et l'affiche.

10.52.11 ovrinos result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_result](#) (int *result_id*, mixed *field*)

[PHP 4 >= 4.0.3]

[ovrimos_result\(\)](#) sert à lire le contenu d'une colonne.

[ovrimos_result\(\)](#) lit le contenu de la colonne *field* dans le résultat *result_id*. *field* peut être le nom de la colonne (une chaîne) ou bien le numéro de la colonne (la première colonne est alors 1).

10.52.12 [ovrimos_result_all](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_result_all](#) (int *result_id*, string *format*)

[PHP 4 >= 4.0.3]

[ovrimos_result_all\(\)](#) sert à afficher tout le résultat d'une requête.

[ovrimos_result_all\(\)](#) affiche le résultat de la requête représentée par *result_id*. Retourne TRUE ou FALSE.

Prépare une requête, l'exécute, et affiche le résultat

```
<?php
$conn = ovrimos_connect ("db_host", "8001", "admin", "password");
if ($conn != 0) {
    echo "Connection établie!";
    $res = ovrimos_prepare ($conn, "select table_id, table_name
from sys.tables where table_id = 7");
    if ($res != 0) {
        echo "Préparation faite!";
        if (ovrimos_execute ($res, array(3))) {
            echo "Exécution réussie!\n";
            ovrimos_result_all ($res);
        } else {
            echo "Exécution manquée!";
        }
    }
    ovrimos_free_result ($res);
} else {
    echo "Préparation manquée!\n";
}
ovrimos_close ($conn);
}
?>
```

Cet exemple exécute une requête SQL et affiche le résultat sous forme d'une table HTML.

ovrimos_result_all() avec meta-information

```
<?php
$conn = ovrimos_connect ("db_host", "8001", "admin", "password");
if ($conn != 0) {
    echo "Connection établie!";
    $res = ovrimos_exec ($conn, "select table_id, table_name
from sys.tables where table_id = 1")
    if ($res != 0) {
        echo "Requête effectuée! cursor=".ovrimos_cursor ($res)."\n";
        $colnb = ovrimos_num_fields ($res);
        echo "Output columns=".$colnb."\n";
        for ($i=1; $i<=$colnb; $i++) {
            $name = ovrimos_field_name ($res, $i);
            $type = ovrimos_field_type ($res, $i);
            $len = ovrimos_field_len ($res, $i);
            echo "Colonne ".$i." nom=".$name." type=".$type." longueur=".$len."\n";
        }
    }
}
```

```
ovrimos_result_all ($res);
ovrimos_free_result ($res);
    }
ovrimos_close ($conn);
}
?>
```

Exemple avec ovrimos_result_all()

```
<?php
$conn = ovrimos_connect ("db_host", "8001", "admin", "password");
if ($conn != 0) {
    echo "Connection établie!";
    $res = ovrimos_exec ($conn, "update test set i=5");
    if ($res != 0) {
        echo "Requête effectuée!";
        echo ovrimos_num_rows ($res). " lignes affectées\n";
        ovrimos_free_result ($res);
    }
    ovrimos_close ($conn);
}
?>
```

10.52.13 ovrimos_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_num_rows](#) (int *result_id*)

[PHP 4 >= 4.0.3]

[ovrimos_num_rows\(\)](#) retourne le nombre de lignes affectées par une modification

10.52.14 ovrimos_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_num_fields](#) (int *result_id*)

[PHP 4 >= 4.0.3]

[ovrimos_num_fields\(\)](#) indique le nombre de colonnes du résultat *result_id*.

10.52.15 ovrimos_field_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_name](#) (int *result_id*, int *field_number*)

[PHP 4 >= 4.0.3]

[ovrimos_field_name\(\)](#) sert à obtenir le nom d'une colonne.

[ovrimos_field_name\(\)](#) retourne le nom d'une colonne à partir de son numéro de colonne *field_number*, (la première colonne est à 1).

10.52.16 ovrimos_field_type

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_type](#)(int *result_id*, int *field_number*)

[PHP 4 >= 4.0.3]

[ovrimos_field_type\(\)](#) sert à connaître le type numérique d'une colonne.

[ovrimos_field_type\(\)](#) retourne le type numérique d'une colonne, identifiée par son numéro *field_number* dans le résultat *field_number*.

10.52.17 ovrimos_field_len

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_len](#)(int *result_id*, int *field_number*)

[PHP 4 >= 4.0.3]

[ovrimos_field_len\(\)](#) sert à connaître la taille d'une colonne.

[ovrimos_field_len\(\)](#) retourne la taille de la colonne *field_number*, dans le résultat *field_number*.

10.52.18 ovrimos_field_num

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_field_num](#)(int *result_id*, string *field_name*)

[PHP 4 >= 4.0.3]

[ovrimos_field_num\(\)](#) sert à connaître le numéro de colonne, à partir de son nom.

[ovrimos_field_num\(\)](#) retourne le numéro de la colonne *field_name* (la numérotation commence à 1), dans *result_id*.

10.52.19 ovrimos_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_free_result](#)(int *result_id*)

[PHP 4 >= 4.0.3]

[ovrimos_free_result\(\)](#) sert à effacer un résultat.

[ovrimos_free_result\(\)](#) libère toutes les ressources prises par le résultat *result_id*. Retourne TRUE.

10.52.20 ovrimos_commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_commit](#)(int *connection_id*)

[PHP 4 >= 4.0.3]

[ovrimos_commit\(\)](#) sert à exécuter une transaction.

[ovrimos_commit\(\)](#) exécute la transaction préparée sur la connexion *connection_id*.

10.52.21 ovrimos_rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ovrimos_rollback](#)(int *connection_id*)

[PHP 4 >= 4.0.3]

[ovrimos_rollback\(\)](#) sert à annuler une transaction.

[ovrimos_rollback\(\)](#) annule la transaction préparée sur la connexion *connection_id*.

10.53 Expressions régulières compatibles Perl

[\[Notes en ligne\]](#)

La syntaxe des masques utilisés dans ces fonctions ressemble fort à celle de Perl. Les expressions seront entourées de délimiteurs, slash (/), par exemple. N'importe quel caractère peut servir de délimiteur, tant qu'il n'est pas alphanumérique ou n'est pas un antislash (\). Si un délimiteur doit être utilisé dans l'expression, il faudra l'échapper avec un antislash. Depuis PHP 4.0.4, vous pouvez utiliser les délimiteurs (), {}, [], et <>, comme en Perl.

Le délimiteur final peut être suivi d'options qui affecteront la recherche. [10.53.7 options de recherche](#).

Exemples de masques valides

- /\w+>/
- |(\d{3})-\d+|Sm
- /^(?i)php[34]/
- {^\s+(\s+)?\$}

Exemples de masques invalides

- /href='(.*)' – délimiteur final manquant
- /\w+\s*\w+/J – option 'J' inconnue
- 1-\d3-\d3-\d4| – délimiteur initial manquant

Note : Les expressions régulières Perl sont disponibles depuis la PHP 4 et PHP 3.0.9.

10.53.1 preg_match

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [preg_match](#)(string *pattern*, string *subject*, array *matches*)

[PHP 3 >= 3.0.9, PHP 4]

[preg_match\(\)](#) analyse *subject* pour trouver l'expression *pattern*.

Si *matches* est fourni, il sera rempli par les résultats de la recherche. \$matches[0] contiendra le texte qui

satisfait le masque complet, `$matches[1]` contiendra le texte qui satisfait la première parenthèse capturante, etc..

[preg_match\(\)](#) retourne TRUE si la recherche réussit, et FALSE sinon (notamment en cas d'erreur).

Extraction d'un numéro de page d'une chaîne.

```
<?php
if (preg_match("/page\s+#(\d+)/i", "Aller à la page numéro 9.", $parts))
print "La page suivante est $parts[1]";
else
print "Page introuvable.";
?>
```

Trouve le mot "web"

```
<?php
// \b, dans le masque, indique une limite de mot, de façon à ce que le mot
// "web" uniquement soit repéré, et pas seulement des parties de mots comme
// dans "webbing" ou "cobweb"
if (preg_match ("/\bweb\b/i", "PHP est le meilleur langage de script du web.")) {
print "Un mot a été trouvé.";
} else {
print "Un mot n'a pas été trouvé.";
}
if (preg_match ("/\bweb\b/i", "PHP est le meilleur langage de script pour les webagency.")) {
print "Un mot a été trouvé.";
} else {
print "Un mot n'a pas été trouvé.";
}
?>
```

Lire un nom de domaine dans une URL

```
<?php
// repérer le nom de l'hôte dans l'URL
preg_match("/^(http:\\\\\/)?([^\\/]+)/i",
"http://www.php.net/index.html", $matches);
$host = $matches[2];
// repérer les deux derniers segments du nom de l'hôte
preg_match("/[^\.\\/]+\.[^\.\\/]+$/", $host, $matches);
echo "Le nom de domaine est : ".$matches[0]."\n";
?>
```

Cet exemple va afficher : Le nom de domaine est : php.net Voir aussi [preg_match_all\(\)](#), [preg_replace\(\)](#) et [preg_split\(\)](#).

10.53.2 preg_match_all

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [preg_match_all](#) (string *pattern*, string *subject*, array *matches*, int *order*)
[PHP 3>= 3.0.9, PHP 4]

[preg_match_all\(\)](#) analyse *subject* pour trouver l'expression *pattern* et met les résultats dans *matches*, dans l'ordre spécifié par *order*.

Après avoir trouvé un premier résultat, la recherche continue jusqu'à la fin de la chaîne.
order peut prendre une des deux valeurs suivantes :

PREG_PATTERN_ORDER

- L'ordre est tel que \$matches[0] est un tableau qui contient les résultats qui satisfont le masque complet, \$matches[1] est un tableau qui contient les résultats qui satisfont la première parenthèse capturante, etc..

```
<?php
preg_match_all("|<[^>]+>(.*</[^>]+>|U", "<b>exemple: </b><div align=left>a test</div>", $
print $out[0][0].", ".$out[0][1]."\n";
print $out[1][0].", ".$out[1][1]."\n";
?>
```

Cet exemple va afficher : exemple: gt; , <div align=left>ceci est un test</div> exemple: , ceci est un test Ainsi, \$out[0] est un tableau qui contient les résultats qui satisfont le masque complet, et \$out[1] est un tableau qui contient les balises entre > et <.

PREG_SET_ORDER

- Les résultats sont classés de telle façon que \$matches[0] contient la première série de résultat, \$matches[1] contient la deuxième série de résultat, etc...

```
<?php
preg_match_all("|<[^>]+>(.*</[^>]+>|U", "<b>exemple: </b><div align=left>un test</div>", $
print $out[0][0].", ".$out[0][1]."\n";
print $out[1][0].", ".$out[1][1]."\n";
?>
```

Cet exemple va afficher : exemple: , exemple: <div align=left>un test</div>, un test Dans ce cas, \$matches[0] est la première série de résultat, et \$matches[0][0] contient le texte qui satisfait le masque complet, \$matches[0][1] contient le texte de la première parenthèse capturante, etc... De même, \$matches[1] contient le texte qui satisfait le masque complet, etc...

Si **order** est omis, PREG_PATTERN_ORDER est utilisé par défaut.

Retourne le nombre de résultat qui satisfont le masque complet, ou FALSE en cas d'échec ou d'erreur.

Extraction de tous les numéros de téléphone d'un texte.

```
<?php
preg_match_all("/\((? (\d{3})? \))? (? (1) [\-\s] ) \d{3}-\d{4}/x",
    "Appelez 555-1212 ou 1-800-555-1212", $phones);
?>
```

Recherche les couples de balises HTML (gourmand)

```
<?php
// Cet exemple utilise les références arrières (\\2).
// Elles indiquent à l'analyseur qu'il doit trouver quelquechose qu'il
// a déjà repéré un peu plus tôt (ici, ([\w]+)).
$html = "<B>Texte en gras</B><a href=salut.html>clicque moi</?>";
preg_match_all ("/(<([\w]+)[?>]?>)(.*<\/\\?>)/", $html, $matches);
for ($i=0; $i< count($matches[0]); $i++) {
```

```

echo "trouvé: ".$matches[0][$i]."\n";
echo "partie 1: ".$matches[1][$i]."\n";
echo "partie 2: ".$matches[3][$i]."\n";
echo "partie 3: ".$matches[4][$i]."\n\n";
}
?>

```

Cet exemple va produire : trouvé: bold text</?> partie 1: partie 2: Test en gras partie 3: trouvé: clique moi</?> partie 1: partie 2: clique moi partie 3: </?>

Voir aussi [preg_match\(\)](#), [preg_replace\(\)](#) et [preg_split\(\)](#).

10.53.3 preg_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [preg_replace](#) (mixed *pattern*, mixed *replacement*, mixed *subject*, int *limit*)
[PHP 3>= 3.0.9, PHP 4]

[preg_replace\(\)](#) analyse *subject* pour trouver l'expression *pattern* et remplace les résultats par *replacement*. *replacement* peut contenir des références de la forme `\n` ou, depuis PHP 4.0.4) `$n`. Cette dernière forme est recommandée. Ces références seront remplacées par le texte capturé par la *n*-ième parenthèse capturante du masque. *n* peut prendre des valeurs de 0 à 99, et `\0` ou `$0`, correspondent au texte de qui satisfait le masque complet. Les parenthèses ouvrantes sont comptées de gauche à droite (en commençant à 1) pour déterminer le numéro de parenthèse capturante.

Si la recherche n'aboutit à aucun résultat, *subject* sera inchangé.

Tous les paramètres de [preg_replace\(\)](#) peuvent être des tableaux.

Si *subject* est un tableau, alors l'opération sera appliquée à chacun des éléments du tableau, et le tableau sera retourné.

Si *pattern* et *replacement* sont des tableaux, alors [preg_replace\(\)](#) prend une valeur de chaque tableau, et l'utilise pour faire la recherche et le remplacement. Si *replacement* a moins d'éléments que *pattern*, alors la chaîne vide est utilisé pour le reste des valeurs. Si *pattern* est un tableau, et que *replacement* est une chaîne, alors cette chaîne sera utilisée pour chaque valeur de *pattern*. Le contraire n'aurait pas de sens.

/e force [preg_replace\(\)](#) à traiter *replacement* comme du code PHP une fois que les substitutions adéquates ont été faites. Conseil : assurez vous que *replacement* est un code PHP valide, car sinon, PHP trouvera une erreur d'analyse (parse error) dans cette ligne.

/F indique que le paramètre *replacement* doit être considéré comme un nom de fonction. Cette fonction sera appelée, avec un tableau contenant les éléments trouvés comme arguments. La fonction doit retourner la chaîne de remplacement. Cette option a été ajoutée en PHP 4.0.4.

Remplacement de plusieurs valeurs

```

<?php
$patterns = array ( "/(19|20)(\d{2})-(\d{1,2})-(\d{1,2})/",
                    "/^s*(\w+)\s*=/" );
$replace = array ( "\\3/\\4/\\1\\2", "$\\1 =");
print preg_replace ($patterns, $replace, "{startDate} = 1999-5-27");
?>

```

Cet exemple va afficher : \$startDate = 5/27/1999

Utilisation de l'option /e

```

<?php

```

```
preg_replace("/(<\/?)(\w+)([^\>]*>/e", "'\\1'.strtoupper('\\2').'\\3'", $html_body);
?>
```

Cela va mettre en majuscule toutes les balises HTML du texte.

Conversion HTML en texte

```
<?php
// $document contient un document HTML
// Ce script va effacer les balises HTML, les javascript
// et les espaces. Il remplace aussi quelques entités HTML
// courante en leur équivalent texte.
$search = array ("<script[?>]*?>.*?</script>'si", // Supprime le javascript
                 "<[\\\/!]*?[^<38;gt;]*?>'si", // Supprime les balises HTML
                 "'([\\r\\n])([\\s]+)'", // Supprime les espaces
                 "'&(quot|#34);'i", // Supprime les entités HTML
                 "'&(amp|#38);'i",
                 "'&(lt|#60);'i",
                 "'&(gt|#62);'i",
                 "'&(nbsp|#160);'i",
                 "'&(iexcl|#161);'i",
                 "'&(cent|#162);'i",
                 "'&(pound|#163);'i",
                 "'&(copy|#169);'i",
                 "'&#(\\d+);'e"); // Evaluation comme PHP
$replace = array ("",
                 "",
                 "\\1",
                 "\\ ",
                 "&",
                 "<",
                 ">",
                 " ",
                 chr(161),
                 chr(162),
                 chr(163),
                 chr(169),
                 "chr(\\1)");
$text = preg_replace ($search, $replace, $document);
?>
```

Note : Le paramètre ***limit*** a été ajouté à partir de PHP 4.0.1pl2.

Voir aussi [preg_match\(\)](#), [preg_match_all\(\)](#) et [preg_split\(\)](#).

10.53.4 preg_split

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [preg_split](#) (string ***pattern***, string ***subject***, int ***limit*** , int ***flags***)
[PHP 3>= 3.0.9, PHP 4]

Note : Le paramètre ***flags*** a été ajouté dans PHP Beta 3.

Retourne un tableau contenant les sous chaînes de ***subject***, séparées par les chaînes qui vérifient ***pattern***.

Si ***limit*** est donné, seules les ***limit*** premières chaînes seront retournées.

Si le flag est PREG_SPLIT_NO_EMPTY, alors seul les sous chaînes non nulles seront retournées.

Eclatement d'une chaîne de recherche.

```
<?php
// scinde la phrase grâce aux virgules et espacements
// ce qui inclus les " ", \r, \t, \n et \f
$keywords = preg_split ("/[\s,]+/", "langage hypertexte, programmation");
?>
```

Scinder une chaîne en caractères

```
<?php
$str = 'string';
$chars = preg_split('///', $str, 0, PREG_SPLIT_NO_EMPTY);
print_r($chars);
?>
```

Voir aussi [preg_match\(\)](#), [preg_match_all\(\)](#) et [preg_replace\(\)](#).

10.53.5 preg_quote

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [preg_quote](#) (string *str*, string *delimiter*)
[PHP 3>= 3.0.9, PHP 4]

[preg_quote\(\)](#) ajoute un antislash devant tous les caractères de la chaîne *str*. Cela est très utile si vous avez une chaîne qui va servir de masque, mais qui est générée durant l'exécution.

Si l'argument optionnel *delimiter* est fourni, il sera aussi échappé. Ceci est pratique pour échapper le délimiteur requis par les fonctions PCRE. Le slash / est le délimiteur le plus répandu.

Les caractères spéciaux qui seront échappés :

```
. \ \ + * ? [ ^ ] $ ( ) { } = ! < > | :
```

Protège des caractères spéciaux

```
<?php
$keywords = "$40 pour un g3/400";
$keywords = preg_quote ($keywords, "/");
echo $keywords; // retourne \$40 pour un g3\400
?>
```

Mise en italique d'un mot dans un texte

```
<?php
// Dans cet exemple, preg_quote($word) sert à éviter que les astérisques
// prennent une valeur particulière dans l'expression régulière.
$textbody = "Ce livre est *très* difficile à trouver.";
$word = "*très*";
$textbody = preg_replace ("/".preg_quote($word)."/",
                        "<B>". $word. "</B>",
                        $textbody);
```

?>

10.53.6 preg_grep

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [preg_grep](#)(string *pattern*, array *input*)
[PHP 4]

[preg_grep\(\)](#) retourne un tableau qui contient les éléments de *input* qui satisfont le masque *pattern*. Depuis PHP 4.0.4, le tableau retourné par [preg_grep\(\)](#) est indexé en utilisant les clés issues du tableau *input*. Si ces clés sont inutiles, utilisez la fonction [array_values\(\)](#) sur le tableau retourné par [preg_grep\(\)](#) pour obtenir le comportement traditionnel.

Exemple avec preg_grep()

```
<?php
// recherche les nombres à virgule flottante
preg_grep("/^(\d+)?\.\d+$/", $array);
?>
```

10.53.7 options de recherche

[\[Notes en ligne\]](#)

Les options de PCRE sont listées ci dessous. Les noms entre parenthèses sont les noms internes à PCRE.

i (PCRE_CASELESS)

- Effectue une recherche insensible à la casse.

m (PCRE_MULTILINE)

- Par défaut, PCRE traite la chaîne sujet comme une seule ligne (même si cette chaîne contient des retours chariot). Le méta-caractère "début de ligne" (^) ne sera valable qu'une seule fois, au début de la ligne, et le méta caractère "fin de ligne" (\$) ne sera valable qu'à la fin de la chaîne, ou avant le retour chariot final (à moins que l'option E ne soit mise). C'est le même fonctionnement qu'en Perl. Lorsque cette option est mise, " début de ligne " et " fin de ligne " correspondront alors aux caractères suivant et précédent immédiatement un caractère de nouvelle ligne, en plus du début et de la fin de la chaîne. C'est le même fonctionnement que l'option Perl /m. Si il n'y a pas de caractère de nouvelle ligne "\n" dans la chaîne sujet, ou si il n'y a aucune occurrence de ^ ou \$ dans le masque, cette option ne sert à rien.

s (PCRE_DOTALL)

- Avec cette option, le méta caractère point (.) remplace n'importe quel caractère, y compris les nouvelles lignes. Sans cette option, le caractère point ne remplace pas les nouvelles lignes. Cette option est équivalente à l'option Perl /s. Une classe de caractère négative telle que [^a] acceptera toujours les caractères de nouvelles lignes, indépendamment de cette option.

x (PCRE_EXTENDED)

- Avec cette option, les caractères d'espacement sont ignorés, sauf lorsqu'ils sont échappés, ou à l'intérieur d'une classe de caractère, et tous les caractères entre # non échappés et en dehors d'une classe de caractère, et le prochain caractère de nouvelle ligne sont ignorés. C'est l'équivalent Perl de l'option /x : elle permet l'ajout de commentaires dans les masques compliqués. Notez bien, cependant, que cela ne s'applique qu'aux caractères de données. Les caractères d'espacement ne doivent jamais apparaître dans les séquences spéciales d'un masque, comme par exemple dans la séquence (?{ qui introduit une parenthèse conditionnelle.

e

- Avec cette option, [preg_replace\(\)](#) effectue la substitution normale des références arrières dans la chaîne de remplacement, puis l'évalue comme un code PHP, et utilise le résultat pour remplacer la chaîne de recherche. Cette option ne peut pas être utilisée avec /F.

Seule [preg_replace\(\)](#) utilise cette option. Elle est ignorée par les autres.

F

- Avec cette option, [preg_replace\(\)](#) considère le paramètre de remplacement comme un nom de fonction, qui doit être appelé pour fournir la chaîne de remplacement. La fonction reçoit un tableau d'éléments trouvés. Cette option ne peut pas être utilisée avec l'option /e.

[preg_replace\(\)](#) seulement utilise cette option. Elle est ignorée par les autres fonctions PCRE. Cette option a été ajoutée en PHP 4.0.4.

A (PCRE_ANCHORED)

- Avec cette option, le masque est ancré de force, c'est à dire que le masque doit s'appliquer entre le début et la fin de la chaîne sujet pour être considéré comme trouvé. Il est possible de réaliser le même effet en ajoutant les méta-caractères adéquats, ce qui est la seule manière de le faire en Perl.

E (PCRE_DOLLAR_ENDONLY)

- Avec cette option, le méta-caractère \$ ne sera valable qu'à la fin de la chaîne sujet. Sans cette option, \$ est aussi valable avant une nouvelle ligne, si cette dernière est le dernier caractère de la chaîne. Cette option est ignorée si l'option *m* est mise. Il n'y a pas d'équivalent en Perl.

S

- Lorsqu'un masque est utilisé plusieurs fois, cela vaut la peine de passer quelques instants de plus pour l'analyser et optimiser le code pour accélérer les traitements ultérieurs. Cette option force cette analyse plus poussée. Actuellement, cette analyse n'est utile que pour les masques non ancrés, qui ne commencent pas par un caractère fixe.

U (PCRE_UNGREEDY)

- Cette option inverse la tendance à la gourmandise des expressions régulières. Vous pouvez aussi inverser cette tendance au coup par coup avec un ?. De même, si cette option est mise, le ? rendra gourmand une séquence. Cette option n'est pas compatible avec Perl. Elle peut aussi être mise dans le masque avec l'option (?U).

X (PCRE_EXTRA)

- Cette option ajoute d'autres fonctionnalités incompatible avec le PCRE de Perl. Tous les antislash suivis d'une lettre qui n'aurait pas de signification particulière cause une erreur, permettant la réservation de ces combinaisons pour des ajouts fonctionnels ultérieurs. Par défaut, comme en Perl, les antislash suivis d'une lettre sans signification particulière sont traités comme des valeurs littérales. Actuellement, cette option ne déclenche pas d'autres fonctions.

10.53.8 syntaxe des masques

[\[Notes en ligne\]](#)

La bibliothèque PCRE est un ensemble de fonctions qui implémentent la recherche par expressions

régulières, en utilisant la même syntaxe et la même sémantique que le Perl 5, avec quelques nuances (voir ci-dessous). L'implémentation actuelle est celle de Perl 5.005.

Les différences avec le Perl 5.005 sont présentées ici :

1. Par défaut, un caractère d'espacement correspond à n'importe quel caractère que la fonction C `isspace()` reconnaît, bien qu'il soit possible de recompiler la bibliothèque PCRE avec d'autres tables de caractères. Normalement, `isspace()` retourne TRUE pour les espaces, les retours chariot, les nouvelles lignes, les formfeed, les tabulations verticales et horizontales. Le Perl 5 n'accepte plus la tabulation verticale comme caractère d'espacement. La séquence `\v` qui était dans la documentation Perl depuis longtemps n'a jamais été reconnue. Cependant, la tabulation verticale elle-même était reconnue comme un caractère d'espacement jusqu'à la version 5.002. Avec les versions 5.004 et 5.005, l'option `\s` l'ignore.
2. PCRE ne tolère pas la répétition de quantificateurs dans les expressions. Perl le permet, mais cela ne signifie pas ce que vous pourriez penser. Par exemple, `(?!a){3}` ne s'interprète pas : les trois caractères suivants ne sont pas des "a". En fait, cela s'interprète comme : le caractère suivant n'est pas "a" trois fois.
3. Les occurrences de sous-masques qui interviennent dans des assertions négatives sont comptées, mais elles ne sont pas enregistrées dans le vecteur d'occurrences. Perl modifie ses variables numériques pour toutes les occurrences de sous-masque, avant que l'assertion ne vérifie le masque entier, et uniquement si les sous-masques ne trouvent qu'une seule occurrence.
4. Bien que les caractères nul soient tolérés dans la chaîne de recherche, ils ne sont pas acceptés dans le masque, car le masque est utilisé comme une chaîne C standard, terminée par le caractère nul. Il faut donc utiliser la séquence d'échappement `"\0"` dans le masque pour rechercher les caractères nul.
5. Les séquences d'échappement suivantes ne sont pas supportées par le Perl : `\l`, `\u`, `\L`, `\U`, `\E`, `\Q`. En fait, elles sont implémentées par la gestion intrinsèque de chaînes du Perl, et ne font pas partie de ses caractères spéciaux.
6. L'assertion `\G` du Perl n'est pas supportée car elle n'est pas pertinente pour faire des recherches avec des masques uniques.
7. De manière assez évidente, PCRE n'accepte pas la construction `(?{code})`.
8. Au moment de l'écriture de PCRE, Perl 5.005_02 avait quelques comportements étranges avec la capture des chaînes lorsqu'une partie du masque est redoublée. Par exemple, "aba" avec le masque `/^(a(b)?)+$/` va affecter à \$2 la valeur "b", mais la même manipulation avec "aabbaa" et `/^(aa(bb)?)+$/` laissera \$2 vide. Cependant, si le masque est remplacé par `/^(aa(b(b))?) +$/` alors \$2 (et d'ailleurs \$3) seront correctement affectés. Avec le Perl 5.004, \$2 sera correctement affecté dans les deux cas, et c'est aussi vrai avec PCRE. Si Perl évolue vers un autre comportement cohérent, PCRE s'adaptera probablement.
9. Une autre différence encore non résolue est le fait qu'en Perl 5.005_02 le masque `/^(a)?(?(1)a|b)+$/` accepte la chaîne "a", tandis que PCRE ne l'accepte pas. Cependant, que ce soit avec Perl ou PCRE `/^(a)?a/` et "a" laisseront \$1 vide.
10. PCRE propose quelques extensions aux expressions régulières du Perl.
 1. (a) Bien que les assertions avec retour (lookbehind) soit obligée d'apparier une chaîne de longueur fixe, toutes les assertions avec retour peuvent avoir une longueur différente. Perl 5.005 leur impose d'avoir toutes la même longueur.
 2. (b) Si PCRE_DOLLAR_ENDONLY est mis, et que PCRE_MULTILINE n'est pas mis, le méta caractère \$ ne s'applique qu'à la fin physique de la chaîne, et non pas avant les caractères de nouvelle ligne.
 3. (c) Si PCRE_EXTRA est mis, un antislash suivi d'une lettre sans signification spéciale est considérée comme une erreur.

4. (d) SI PCRE_UNGREEDY est mis, la "gourmandise" des quantificateurs de répétition est inversée, ce qui est rend non gourmand par défaut, mais si ils sont suivis de ?, il seront gourmands.

La syntaxe et la sémantique des expressions régulières supportées par PCRE sont décrites ci-dessous. Les expressions régulières sont aussi décrites dans la documentation Perl, et dans un grand nombre d'autres livres, avec de nombreux exemples. Jeffrey Friedl's "Mastering Regular Expressions", édité chez O'Reilly (ISBN 1-56592-257-3), les décrits en profondeur. Cette description est organisée comme une documentation de référence.

Une expression régulière est un masque, qui est appliqué sur une chaîne sujet, de gauche à droite. La plus part des caractères se représentent eux-mêmes. Un exemple trivial : un masque qui serait Le rapide renard gris

Pourra correspondre à une partie de la chaîne sujet qui sera identique au masque. La puissance des expressions régulières provient de leur capacité à autoriser des alternatives et des quantificateur de répétitions dans le masque. Ils sont encodés dans le masque par des *meta-characters*, qui ne représentent pas ce qu'ils sont, mais sont interprétés d'une certaine manière.

Il y a deux sortes de méta-caractères : ceux qui sont reconnus n'importe où dans un masque, hormis entre crochets, et ceux qui sont reconnus entre crochets. A l'extérieur des crochets, les méta caractères sont :

\	Caractère d'échappement, avec de multiples usages.
^	Le début de la chaîne sujet (ou de ligne, en mode multiligne)
\$	La fin de la chaîne sujet (ou de ligne, en mode multiligne)
.	Remplace n'importe quel caractère, hormis le caractère de nouvelle ligne (par défaut) ;
[Caractère de début de définition de classe
]	Caractère de fin de définition de classe
	Caractère de début d'alternative
(Caractère de début de sous-masque
)	Caractère de fin de sous-masque
?	Etend le sens de (

mais aussi quantificateur de 0 ou 1

mais aussi quantificateur de minimisation

*	Quantificateur de 0 ou plus
+	Quantificateur de 1 ou plus
{	Caractère de début de quantificateur minimum/maximum

La partie du masque qui est entourée de crochet et appelé une classe de caractères. Dans les classes

\	Caractère d'échappement, avec de multiples usages
^	négation de la classe, mais uniquement si placé tout au début de la

classe

-	indique un intervalle de caractères
]	termine la classe de caractères

La section suivante décrit l'utilisation de chaque méta caractères :

ANTISLASH

Le caractère antislash a de nombreuses utilisations. En premier lieu, s'il est suivi d'un caractère non alpha numérique, il ne prendra pas la signification spéciale qui y est rattachée. Cette utilisation de l'antislash comme caractère d'échappement s'applique à l'intérieur et à l'extérieur des classes

de caractères. Par exemple, pour recherche le caractère

étoile "*", il faut écrire dans le masque : "*". Cela s'applique dans tous les cas, que le caractère qui suive

soit un méta-caractère ou non. C'est un moyen sûr pour s'assurer qu'un caractère sera recherché pour sa valeur littérale, plutôt que pour sa valeur spéciale. En particulier, pour rechercher les antislash, il faut écrire : "\\". Si un masque est utilisé avec l'option PCRE_EXTENDED, les espaces blancs du masque, mais qui ne sont pas dans une classe de caractères, et les caractères entre dièses "#", ainsi que les nouvelles lignes sont ignorées. L'antislash peut être utilisé pour échapper et ainsi rechercher un espace ou un dièse. La deuxième utilité de l'antislash est de pouvoir coder des caractères invisibles dans les masques. Il n'y a pas de restriction sur la place de ces caractères invisibles, hormis pour le caractère nul qui doit terminer le masque. Lors de la préparation du masque, il est souvent plus pratique d'utiliser les séquences d'échappement suivantes, plutôt que le caractère binaire qu'elle représente :

```
\a    alarme, c'est à dire le caractère BEL (hex 07)
\cx    "control-x", avec x qui peut être n'importe quel caractère.
\e    escape (hex 1B)
\f    formfeed (hex 0C)
\n    nouvelle ligne (hex 0A)
\r    retour chariot (hex 0D)
\t    tabulation (hex 09)
\xhh   caractère en hexadécimal, de code hh
\ddd   caractère en octal, de code ddd, ou référence arrière
```

Dans la séquence "\cx" si "x" est en minuscule, il est converti en majuscule. Puis, le bit 6 (hex 40) est inversé. Ainsi "\cz" devient 1A, mais "\c{" devient hex 3B, tandis que "\c;" devient hex 7B.

Après "\x", deux caractères hexadécimaux sont lus (les lettres peuvent être en majuscule ou minuscule).

Après "\0", deux caractères octal sont lus. Dans chacun des cas, le méta-caractère tente de lire autant de caractère que possible. Ainsi la séquence "\0\x07", sera comprise comme deux caractères nuls, suivi d'un caractère alarme (BEL). Assurez vous que vous fournissez suffisamment de chiffres après le méta-caractère.

La gestion de la séquence "\y", avec y <> 0 est plutôt compliquée. En dehors des caractères de classes, PCRE va lire y et tous les caractères qui suivent comme des chiffres décimaux. Si y est plus petit que 10, ou bien si il y a déjà eu au moins autant de parenthèses ouvrantes auparavant, la séquence est prise pour une *référence de retour*. Le détail sera vu ultérieurement, après la section sur les sous-masques.

A l'intérieur d'un caractère de classe, ou si y est plus grand que 10, et qu'il n'y a pas eu assez de parenthèses ouvrantes auparavant, PCRE lis jusqu'à 3 chiffres octals à la suite de l'antislash, et génère un octet unique, à partir des 8 bits de poids faible de la séquence. Tous les chiffres qui suivent ne sont pas interprétés, et se représentent eux-mêmes. Par exemple,

```
\040  est une autre manière d'écrire un espace
\40    est identique, dans la mesure où il n'y a pas 40 parenthèses
ouvrantes auparavant.
\7      est toujours une référence de retour.
\11     peut être une référence de retour, ou une tabulation, tandis que
\011    est toujours une tabulation
\011    est toujours une tabulation
\0113   est une tabulation suivi du caractère "3"
\113    est le caractère 113 (étant donné qu'il ne peut y avoir plus de
99 référence de arrière)
\377    est un octet dont tous les bits sont à 1
\81     peut être soit une référence de arrière, soit le caractère NULL, suivi des
caractères "8" et "1"
```

Les valeurs octales supérieures ou égales à 100 ne doivent pas être introduite par un 0, car seuls les trois premiers octets seront lus.

Toutes les séquences qui définissent une valeur d'un seul octet peuvent être

utilisé dans les classes de caractères, et à l'extérieur. De plus, dans une classe de caractère, la séquence "\b" est interprétée comme un caractère effacer (backspace, hex 08). A l'extérieur d'une classe de caractères, il peut avoir d'autres significations (voir ci-dessous).

On peut encore se servir de l'antislash pour préciser des types génériques de valeurs :

```
\d    tout caractère décimal
\D    tout caractère qui n'est pas un caractère décimal
\s    tout caractère blanc
\S    tout caractère qui n'est pas un caractère blanc
\w    tout caractère de "mot"
\W    tout caractère qui n'est pas un caractère de "mot"
```

Chaque paire précédente définit une partition de la table des caractères : les deux ensembles sont disjoints. Un caractère satisfera soit un méta-caractère, soit l'autre.

Un caractère de "mot" sera une lettre, un chiffre ou le caractère souligné, c'est à dire un caractère qui pourra être une partie d'un mot Perl. La définition des lettres et chiffres est définie par les tables de caractères de PCRE, et peut varier suivant la table locale de caractère (voir "Tables de caractères locales ", ci-dessus. Par exemple, dans la configuration français ("fr"), certains caractères ont des codes supérieurs à 128, pour les caractères accentués, et ils seront compris par le méta caractère \w.

Ces séquences de caractères peuvent apparaître à l'intérieur ou à l'extérieur des classes de caractères. Elles remplacent à chaque fois un caractère du type correspondant. Si cette séquence est mis en fin de masque, et qu'il n'y a plus de caractère à comparer dans la chaîne sujet, la recherche échoue.

La quatrième utilisation de l'antislash intervient lors d'assertions simples.

Une assertion impose une condition à un certain point, sans remplacer de caractère. L'utilisation de sous-masques pour réaliser des assertions plus complexes est décrites plus bas. Les assertions avec antislash sont les suivantes :

```
\b    limite de mot
\B    pas limite de mot
\A    début de la chaîne sujet (indépendant du mode multi-lignes)
\Z    fin de la chaîne sujet ou nouvelle ligne à la fin de la chaîne sujet
      (indépendant du mode multi-lignes)
\z    fin de la chaîne sujet (indépendant du mode multi-lignes)
```

Ces assertions ne peuvent pas apparaître dans une classe de caractère

(mais "\b" a une autre signification à l'intérieur d'une classe de caractères).

Une limite de mot est un emplacement dans la chaîne sujet ou un caractère et son suivant ne sont pas en même temps des caractères de mot, ou le contraire

(on peut le voir comme \w\W ou \W\w), ou encore le premier ou le dernier caractère est un caractère mot.

Les assertions \A, \Z, et \z diffèrent des méta caractères ^et \$ dans la mesure où ils ne sont pas dépendants des options, notamment PCRE_NOTBOL ou PCRE_NOTEOL.

La différence entre \Z et \z tient au fait que \Z recherche les positions avant les nouvelles lignes et à la fin de la chaîne sujet, tandis que \z ne recherche que la fin de la chaîne.

CIRCUMFLEX et DOLLAR

En dehors d'une classe de caractère, avec les options par défaut,

^ est une assertion qui n'est vraie que si elle est placée tout au début de la chaîne. A l'intérieur d'une classe de caractère, ^a un tout autre sens (voir ci-dessous).

^ n'a pas besoin d'être le premier caractère du masque, si plusieurs alternatives sont proposées, mais il doit être placé en premier dans chaque alternative.

Si toutes les alternatives commencent par ^, alors le masque est dit ancré

(il y a une autre construction qui porte cette appellation).

\$ est une assertion qui n'est vraie que si elle est placée tout en fin de chaîne ou juste avant un caractère de nouvelle ligne qui serait le dernier caractère de la chaîne. A l'intérieur d'une classe de caractère, ^a un tout autre sens (voir ci-dessous).

\$ n'a pas besoin d'être le dernier caractère du masque, si plusieurs alternatives

sont proposées, mais il doit être placé en dernier dans chaque alternative. Si toutes les alternatives finissent par \$, alors le masque est dit ancré (il y a une autre construction qui porte cette appellation). \$ n'a pas de valeur particulière dans une classe de caractères.

La signification de \$ peut changer, de manière à l'amener à ce qu'il ne puisse se trouver qu'en toute fin de la chaîne sujet. Cela se fait en ajoutant l'option PCRE_DOLLAR_ENDONLY au moment de la compilation, ou de l'exécution. Cette option est inopérante sur \Z.

La signification de ^ peut changer, de manière à l'amener à ce qu'il puisse se trouver immédiatement avant et immédiatement après un caractère de nouvelle ligne "\n". Cela se fait en ajoutant l'option PCRE_MULTILINE au moment de la compilation, ou de l'exécution. Par exemple, le masque /^abc\$/ accepte la chaîne "def\nabc" uniquement en mode multi-lignes. Par conséquent, toutes les parties du masques qui commencent par "^" ne sont pas ancrées, en mode multi ligne. L'option PCRE_DOLLAR_ENDONLY est ignorée si l'option PCRE_MULTILINE est choisie. Notez que les méta caractères \A, \Z, et \z peuvent servir à repérer le début et la fin du sujet, et toutes les parties du masque qui commenceront par \A seront toujours ancrées, avec l'option PCRE_MULTILINE ou non.

FULL STOP (PERIOD, DOT)

En dehors d'une classe de caractères, un point remplace n'importe quel caractère, même invisible et à l'exception du caractère de nouvelle ligne. Avec l'option PCRE_DOTALL le point remplace n'importe quel caractère, même le caractère de nouvelle ligne. La gestion des points est complètement indépendante de ^ et \$. Le seul point commun est que les deux ont un comportement particulier vis à vis des caractères de nouvelle ligne. Le point n'a pas de comportement particulier dans une classe de caractères.

SQUARE BRACKETS

Un crochet ouvrant introduit une classe de caractère, et le crochet fermant la conclut. Le crochet fermant n'a pas de signification en lui même. Si le crochet fermant est nécessaire à l'intérieur d'une classe de caractères, il faut qu'il soit le premier caractère (après un ^ éventuel) ou échappé avec un antislash. Une classe de caractère remplace un seul caractère dans la chaîne sujet, à moins que le premier caractère de la classe soit un ^, qui représente une négation : le caractère ne doit pas se trouver dans la classe. Si ^ est nécessaire dans la classe, il suffit qu'il ne soit pas le premier caractère, ou bien qu'il soit échappé avec un antislash.

Par exemple, le caractère [aeiou] remplace n'importe quelle voyelle minuscule, tandis que [^aeiou] remplace n'importe quelle caractère qui n'est pas une voyelle minuscule. ^ est une notation pratique pour spécifier des caractères qui sont dans une classe, en ne citant que ceux qui n'y sont pas. Le comportement est inchangé.

Avec l'option d'insensibilité à la casse, toutes les lettres d'une classe de caractère représentent en même temps la majuscule et la minuscule. Par exemple, [aeiou] représentera "A" ou "a", et [^aeiou] n'acceptera pas "A", tandis que sans l'option, elle l'accepterait.

Le caractère de nouvelle ligne n'est pas traité de manière spéciale dans les classes de caractère, quelque soit l'option PCRE_DOTALL ou PCRE_MULTILINE.

Une classe telle que [^a] acceptera toujours une nouvelle ligne.

Le signe moins (-) est utilisé pour spécifier un intervalle de caractères, dans une classe. Par exemple, [d-m] remplace toutes les lettres entre d et m inclus. Si le caractère moins est requis dans une classe, il faut l'échapper avec un antislash, ou le faire apparaître à une position où il ne pourra pas être interprété comme une indication d'intervalle, c'est à dire au début ou à la fin de la classe.

Il n'est pas possible d'avoir le caractère "]" comme fin d'intervalle. Un masque tel que [W-]46] est compris comme la classe de caractère contenant deux caractères ("W" et "-") suivi de la chaîne littérale "46]", ce qui fait qu'il va accepter "W46]" ou "-46]". Cependant, si "]" est échappé avec un antislash, le masque [W-\]46] est interprété comme une classe d'un seul caractère, contenant un intervalle de caractère. La valeur octale ou hexadécimale de "]" peuvent aussi être utilisée pour déterminer les limites de l'intervalle.

Les intervalles travaillent sur des séquences ASCII. Elles peuvent aussi être

précisées avec des valeurs numériques, par exemple `[\000-\037]`. Si cet intervalle inclus des lettres utilisées avec une option d'insensibilité de casse, les majuscules ou minuscules correspondantes seront aussi incluses. Par exemple,

`[W-c]` est équivalent à `[[\^_`wxyzabc]`, avec l'option d'insensibilité de casse. Si la table locale de caractère est "fr", `[\xc8-\xcb]` correspond aux caractères accentués.

Les types de caractères `\d`, `\D`, `\s`, `\S`, `\w`, et `\W` peuvent aussi intervenir dans les classes de caractères. Par exemple, `[\dABCDEF]` acceptera n'importe quel caractère hexadécimal. Un accent circonflexe peut aussi être utilisé pour spécifier adroitement des ensembles de caractères plus restrictifs : par exemple `[\^W_]` accepte toutes les lettres et les chiffres, mais pas les soulignés.

Tous les caractères non alphanumériques autres que `\`, `-`, `^` (placé en début de chaîne) et `]` n'ont pas de significations particulière, mais ils ne perdront rien à être échappés.

VERTICAL BAR

La barre verticale sert à séparer des alternatives. Par exemple, dans le masque `dupont|martin`

recherche soit "dupont", soit " martin ". Le nombre d'alternative n'est pas limité, et il est même possible d'utiliser la chaîne vide. Lors de la recherche, toutes les alternatives sont essayées, de gauche à droite, et la première qui est acceptée, est utilisée. Si les alternatives sont dans un sous-masque, elle ne réussiront que si le masque principal réussit aussi.

INTERNAL OPTION SETTING

Les options `PCRE_CASELESS`, `PCRE_MULTILINE`, `PCRE_DOTALL`, et `PCRE_EXTENDED` peuvent être changée depuis le masque lui-même, avec des séquences mises "(?" et ")".

Les options sont

i pour `PCRE_CASELESS`

m pour `PCRE_MULTILINE`

s pour `PCRE_DOTALL`

x pour `PCRE_EXTENDED`

Par exemple, `(?im)` rend le masque insensible à la casse, et multi-lignes. Il est possible d'annuler ces options en les faisant précéder par un signe `-` : par exemple `(?im-sx)`, ajoutera les options `PCRE_CASELESS` et `PCRE_MULTILINE` mais annulera les options `PCRE_DOTALL` et `PCRE_EXTENDED`. Si une option apparaît avant et après le signe moins, l'option sera annulée.

Le domaine d'application de ces options dépend de la position de la séquence d'option. Pour toutes les séquences d'options qui sont hors des sous-masques

(définis plus loin), l'effet est le même que si l'option avait été fixée dès le début de la recherche. Les exemples suivants se comportent tous de la même façon :

`(?i)abc`

`a(?i)bc`

`ab(?i)c`

`abc(?i)`

et sont parfaitement équivalents au masque `abc` avec l'option `PCRE_CASELESS`.

En d'autres termes, mettre des séquences d'options dans le corps principal du masque revient à appliquer l'option à tout le masque, sauf ordre contraire dans les sous-masques. Si il y a plusieurs séquences d'option qui portent sur la même option, la dernière s'appliquera.

Si une option intervient dans un sous-masque, le comportement est différent.

C'est un changement de comportement apparu en Perl 5.005. Une option à l'intérieur d'un sous-masque n'affecte que cette partie du masque, ce qui fait que

`(a(?i)b)c`

acceptera `abc` et `aBc` mais aucune autre chaîne (en supposant que `PCRE_CASELESS` n'est pas utilisé). Cela signifie que les options permettent d'avoir différente configuration de recherche pour différentes parties du masque. Une séquence d'option dans une alternative affecte toute l'alternative. Par exemple :

`(a(?i)b|c)`

accepte "ab", "aB", "c", et "C", même si, comme dans le cas de "C", la première alternative qui porte l'option n'est pas prise en compte. Sinon, cela risque d'introduire des comportements très étranges :

Les options spécifiques à PCRE telles que `PCRE_UNGREEDY` et `PCRE_EXTRA` peuvent

être modifiées de la même manière, en utilisant respectivement les caractères U et X. L'option (?X) est particulière, car elle doit toujours intervenir avant toutes les autres options, même au niveau du masque entier. Il vaut mieux la mettre au début du masque.

sous-masques

Les sous-masques sont délimités par des parenthèses, et peuvent être imbriqués. Ajouter des sous-masques a deux utilités :

1. Délimiter des alternatives. Par exemple, le masque

```
cat(aract|erpillar|)
```

acceptera les mots "char", "chardon", ou "charmant". Sans les parenthèses, il n'accepterait que "chardon", "mant" ou la chaîne vide.

2. Le sous-masque est considéré comme capturant : Lorsqu'une

chaîne sujet est acceptée par le masque complet, les sous masques sont transmis à l'appelant grâce à un vecteur de sous-masques. Les parenthèses ouvrantes sont comptées de gauche à droite, (commençant à 1).

Par exemple, soit la chaîne sujet "le roi soleil " qui est utilisée avec le masque suivant :

```
Le ((roi|prince) (soleil|charmant))
```

les sous-masques capturés sont "roi soleil ", "roi", et "soleil", numérotés respectivement 1, 2, et 3.

L'ubiquité des parenthèses n'est pas toujours simple

d'emploi. Il y a des moments où regrouper des sous-masques est nécessaire, sans pour autant capturer

la valeur trouvée. Si une parenthèse ouvrante est suivie de "?:", le sous-masque ne capture pas la chaîne assortie, et ne sera pas compté lors de la numérotation des captures. Par exemple, avec la chaîne "le prince charmant", utilisé avec le masque pattern

```
Le (( ?roi|prince) (soleil|charmant))
```

les chaînes capturées seront "prince charmant " et "charmant", numérotés respectivement 1 et 2. Le nombre maximal de chaîne capturées est de 99, et le nombre total de sous-masque (capturant ou non) ne doit pas dépasser 200.

```
(?i:samedi|dimanche)
```

```
(?:(?i) samedi | dimanche)
```

De plus, comme les séquences d'options sont valables sur toute une alternative, le masque ci dessus acceptera aussi "DIMANCHE" que "Dimanche".

REPETITION

les Répétitions sont spécifiées avec des quantificateurs, qui peuvent être placés à la suite des caractères suivants :

Un caractère unique, même s'il s'agit d'un méta caractère

Une classe de caractères

Une référence de retour (Voir section suivante)

Un sous-masque avec parenthèses (a moins que ce ne soit une assertion, voir plus loin)

Les quantificateurs généraux précisent un nombre minimum et maximum de répétitions

possibles, donnés par deux nombres entre accolades, et séparés par une virgule.

Ces nombres doivent être plus petit que 65536, et le premier nombre doit être

égal ou inférieur au second. Par exemple

```
z{2,4}
```

accepte "zz", "zzz", ou "zzzz". L'accolade fermante n'a pas de signification par elle même. Si le second nombre est omis, mais que la virgule est là, cela signifie qu'il n'y a pas de limite supérieure. Si le second nombre et la virgule sont omis, le quantificateur correspond au nombre exact de répétition attendues.

Par exemple :

```
[aeiou]{3,}
```

accepte n'importe quelle succession d'au moins 3 voyelles minuscules, tandis que

```
\d{8}
```

n'accepte que 8 chiffres exactements. Une accolade ouvrante qui apparaît à une position où un quantificateur n'est pas accepté, ou si la syntaxe des

quantificateurs n'est pas respectée, elle sera considérée littérale. Par exemple,

```
{,6}
```

n'est pas un quantificateur, mais une chaîne de 4 caractères.

Le quantificateur {0} est autorisé, mais l'expression est alors ignorée.

* équivalent à {0,}

+ équivalent à {1,}

? équivalent à {0,1}

Il est possible de constituer des boucles infinies en créant un sous-masque sans caractères, mais pourvu d'un quantificateur sans limite supérieure.

Par exemple

```
(a?)*
```

Les versions plus anciennes de Perl et PCRE génèrent alors une erreur au moment de la compilation. Cependant, étant donné qu'il existe des situations où ces constructions peuvent être utiles, ces masques sont désormais autorisés. Cependant, si la répétition du sous-masque ne trouve aucun caractère, la boucle est interrompue.

Par défaut, les quantificateurs sont "gourmands", c'est à dire, qu'ils cherchent d'abord à trouver le nombre maximal de répétitions qui autorise le succès de la recherche. L'exemple classique posé par cette gourmandise est la recherche de commentaire d'un programme en C. Les commentaires apparaissent entre les séquences /* et */ et à l'intérieur de ces délimiteurs, les * et / sont autorisés.

Appliquer le masque

```
/\*.*/
```

à la chaîne

```
/* first comment */ not comment /* second comment */
```

ne peut réussir, car le masque travaille sur toute la chaîne, à cause de la gourmandise du caractère .*.

Cependant, un quantificateur suivi d'un point d'interrogation cesse d'être gourmand, et au contraire, ne recherche que le nombre minimum de répétition.

Dans ces conditions, le masque

```
/\*.??*/
```

trouvera bien les commentaires du code C. La signification des autres quantificateurs n'est pas changée. Attention à ne pas confondre l'utilisation du point d'interrogation ici avec son utilisation comme quantificateur lui-même. A cause de cette ambiguïté, il peut apparaître des situations où il faut le doubler :

```
\d??\d
```

Ce masque va tenter de lire un seul chiffre, mais le cas échéant, il acceptera 2 chiffres pour permettre à la recherche d'aboutir.

Si l'option PCRE_UNGREEDY est mise, (une option qui n'est pas disponible avec Perl) alors les quantificateurs sont non gourmand par défaut, mais peuvent être rendu gourmand au cas par cas, en ajoutant un point d'interrogation après. En d'autres termes, cette option inverse le comportement par défaut.

Lorsqu'un sous-masque est quantifié avec un nombre minimum de répétition, qui soit plus grand que 1, ou avec un maximum de répétition, le masque compilé aura besoin de plus de place de stockage, proportionnellement au minimum et au maximum.

Si un masque commence par .* ou {0,} et que l'option PCRE_DOTALL (équivalent en Perl à /s) est mise, c'est à dire en autorisant le remplacement des nouvelles lignes par un méta caractère, alors le masque est implicitement ancré, car tout ce qui suit va être mangé par la première séquence, et se comportera comme si le masque se terminait par le méta caractère \A. Dans le cas où on sait d'avance qu'il n'y aura pas de caractère de nouvelle ligne, mettre l'option PCRE_DOTALL et commencer le masque par .* permet d'optimiser le masque. Alternativement, on peut utiliser

```
^ pour ancrer explicitement le masque.
```

Lorsqu'un sous-masque capturant est répété, la valeur capturée est la dernière.

Par exemple, après que

```
(inter[net]{3}\s*)+
```

```
*)+
```

ai été appliqué à "internet interne" la valeur de la chaîne capturée est

"interne". Cependant, si il y a des sous-masques imbriqués, la valeur capturée correspondante peut l'avoir été lors des précédentes itérations. Par exemple :

```
/(a|(b))+/
```

accepte "aba" et la deuxième valeur capturée est

Références arrières (back references)

En dehors des classes de caractères, un antislash suivi d'un nombre plus grand

que 0 (et possiblement plusieurs chiffres) est une référence arrière (c'est à dire vers la gauche) dans le masque, en supposant qu'il y ait suffisamment de sous-masques capturant précédent.

Cependant, si le nombre décimal suivant l'antislash est plus petit que 10, il sera toujours considéré comme une référence arrière, et cela générera une erreur si le nombre de capture n'est pas suffisant. En d'autres termes, il faut qu'il existe suffisamment de parenthèses ouvrantes à gauche de la référence, surtout si la référence est inférieure à 10. Reportez vous à la section "antislash" pour avoir de plus amples détails à propos du nombre de chiffres qui suivent l'antislash. La référence arrière remplace ce qui a été capturé par un sous-masque dans le masque courant, plutôt que remplace le sous-masque lui même. Ainsi

```
(calme|rapide) et \lment
```

trouvera "calme et calmement " et "rapide et rapidement ", mais pas "calme et rapidement ". Si la recherche tiens compte de la casse, alors la casse de la chaîne capturée sera importante. Par exemple,

```
((?i)rah)\s+\l
```

trouve "rah rah" et "RAH RAH", mais pas "RAH rah", même si le sous-masque capturant initial ne tenait pas compte de la casse.

Il peut y avoir plusieurs références arrières dans le même sous-masque. Si un sous-masque n'a pas été utilisé dans une recherche, alors les références arrières échoueront. Par exemple

```
(a|(bc))\2
```

ne réussira jamais si la chaîne sujet commence par "a" plutôt que par "bc". Etant donné qu'il peut y avoir jusqu'à 99 références arrières, tous les chiffres après l'antislash sont considérés comme faisant potentiellement partie de la référence arrière. Si le masque recherche un chiffre après la référence, alors il faut impérativement utiliser des délimiteurs pour terminer la référence arrière. Si l'option PCRE_EXTENDED est mise, on peut utiliser un espace. Sinon, un commentaire vide fait l'affaire.

Une référence arrière qui intervient à l'intérieur de parenthèses auquel elle fait référence échouera dès que le sous-masque sera utilisé. Par exemple,

```
(a\1) échouera toujours.
```

Cependant, ces références peuvent être utiles dans les sous-masques répétitifs. Par exemple, le masque

```
(a|b\1)+
```

pourra convenir pour "a", "aba", "ababaa" etc. A chaque itération du sous-masque, la référence arrière utilise le résultat du dernier sous-masque. Pour que cela fonctionne, il faut que la première itération n'ai pas besoin d'utiliser la référence arrière. Cela arrive avec les alternatives, comme dans l'exemple ci dessus, ou avec un quantificateur de minimum 0.

Les assertions

Une assertion est un test sur les caractères suivants ou précédent celui qui est en cours d'étude. Ce test ne consomme pas de caractère (ie, on ne déplace pas le pointeur de caractères). Les assertions simples sont codées avec \b, \B, \A, \Z, \z, ^ et \$, et sont décrite précédemment. Il existe cependant un type d'assertion plus complexe, codées sous la forme de sous-masques. Il en existe deux types : ceux qui travaille au dela de la position courante, et ceux qui travaille en deça.

```
\w+(?=;)
```

Une assertion se comporte comme un sous-masque, hormis le fait qu'elle ne déplace pas le pointeur de position. Les assertions avant commencent par (?= pour les assertions positives et par (?! pour des assertions négatives. Par exemple :

```
foo(?!bar)
```

s'assure qu'un mot est suivi d'un point virgule, mais n'inclus pas le point virgule dans la capture. D'autre part,

```
(?!foo)bar
```

en est proche, mais ne trouve pas une occurrence de "bar" qui soit précédée par quelque chose d'autre que "foo"; il trouve toutes les occurrences de "bar", quelque soit ce qui le précède, car l'assertion (?!foo) est toujours vraie quand les trois caractères suivants sont "bar". Une assertion arrière est ici nécessaire. Ces assertions commencent par (?<= pour les assertions positives, et (?<! pour les assertions négatives. Par exemple :

```
(?<!(foo)bar
```

trouve les occurrences de "bar" qui ne sont pas précédées par "foo". Le contenu d'une référence arrière est limité de tel façon que les chaînes qu'il utilise sont toujours de la même taille. Cependant, lorsqu'il y a plusieurs alternatives, elles n'ont pas besoin d'être de la même taille.

```
(?<=bullock|donkey)
```

est autorisé, tandis que

```
(?<!dogs?|cats?)
```

provoque une erreur de compilation. Les alternatives qui ont des longueurs différentes ne sont autorisées qu'au niveau supérieur des assertions arrières. C'est une amélioration du fonctionnement de Perl 5.005, qui impose aux alternatives d'avoir toutes la même taille. Une assertion telle que

```
(?<=ab(c|de))
```

n'est pas autorisée, car l'assertion de bas niveau (la deuxième, ici) a deux alternatives de longueurs différentes. Pour la rendre acceptable, il faut écrire

```
(?<=abc|abde)
```

L'implémentation des assertions arrières déplace temporairement le pointeur de position vers l'arrière, et cherche à vérifier l'assertion. Si le nombre de caractère est différent, la position ne sera pas correcte, et l'assertion

échouera. La combinaison d'assertions arrières avec des sous-masques peut être particulièrement pratique à fin des chaînes. Un exemple est donné à la fin de cette section.

Plusieurs assertions peuvent intervenir successivement. Par exemple

```
(?<=\d{3})(?!999)foo
```

recherche les chaînes "foo" précédée par trois chiffres qui ne sont pas "999".

Notez que chaque assertions est appliquées indépendemment, au même point de la chaîne à traiter. Tout d'abord, il est vérifié que les trois premiers caractères ont tous des chiffres, puis on s'assure que ces trois caractères ne sont pas "999". Le masque précédant n'accepte pas "foo" précédé de 6 caractères, les trois premiers étant des chiffres et les trois suivant

étant différents de "999". Par exemple, ce masque n'acceptera pas la chaîne "123abcfoo". Pour ce faire, il faut utiliser :

```
(?<=\d{3}...)(?!999)foo
```

Dans ce masque, la première assertion vérifie les six premiers caractères, s'assure que les trois premiers sont des entiers, et la deuxième assertion s'assure que les trois derniers caractères ne sont pas "999".

De plus, les assertions peuvent être imbriquées :

```
(?<=(?!foo)bar)baz
```

recherche les occurrences de "baz" qui sont précédées par "bar", qui, à son tour, n'est pas précédé par "foo". Au contraire,

```
(?<=\d{3}(?!999)...)foo
```

est un autre masque, qui recherche les caractères "foo", précédés par trois chiffres, suivis trois autres caractères qui ne forment pas "999".

Les assertions ne sont pas capturantes, et ne peuvent pas être répétées. Si une assertion contient des sous-masques capturants en son sein, ils seront compris dans le nombre de sous-masques capturants du masque entier. La capture est réalisée pour les assertions positives, mais cela n'a pas de sens pour les assertions négatives.

200 assertions au maximum sont autorisées.

sous-masques uniques (ONCE-ONLY SUBPATTERNS)

Avec les quantificateurs de répétitions, l'échec d'une recherche conduit normalement à une autre recherche, avec un nombre différent de répétitions, pour voir si le masque ne s'applique pas dans d'autres conditions. Parfois, il est pratique d'éviter ce comportement, soit pour changer la nature de la recherche, soit pour la faire abandonner plus tôt, si on pense qu'il n'est pas besoin d'aller plus loin.

Considérons par exemple, le masque `\d+foo` appliqué à la ligne
123456bar

Après avoir tenté d'utiliser les 6 chiffres suivi de "foo" qui fait échouer, l'action habituelle sera de réessayer avec 5 chiffres, puis avec 4, et ainsi de suite jusqu'à l'échec final. Un sous-masque évalué une seule fois permettrait d'indiquer que lorsqu'une partie du masque est trouvée, elle n'a pas besoin

d'être réévaluée à chaque tentative. Ceci conduirait à ce que la recherche échoue immédiatement après le premier test. Ces assertions ont leur propre notation, commençant avec (?>comme ceci :

```
(?>\d+)bar
```

Après avoir tenté d'utiliser les 6 chiffres suivi de "foo" qui fait échouer, l'action habituelle sera de réessayer avec 5 chiffres, puis avec 4, et ainsi de suite jusqu'à l'échec final. Un sous-masque évalué une seule fois permettrait d'indiquer que lorsqu'une partie du masque est trouvée, elle n'a pas besoin d'être réévaluée à chaque tentative. Ceci conduirait à ce que la recherche échoue immédiatement après le premier test. Ces assertions ont leur propre notation, commençant avec (?> comme ceci : (?>\d+)bar. Ce type de parenthèses verrouille le sous-masque qu'il contient un fois qu'il a été trouvé, et empêche un

échec ultérieur d'y repasser, mais autorise à revenir plus loin en arrière. Une autre description est que les sous-masques de ce type recherche les chaînes de caractères, et les ancre le sous-masque à l'intérieur de la chaîne. Les sous-masques uniques ne sont pas capturants. Des cas simples comme ceux présentés ci dessus peuvent être pris comme des situations maximisantes, qui réservent le maximum de caractères. En effet, alors que \d+ et \d+? ajustent le nombre de chiffres trouvés de manière à laisser la possibilité au masque de réussir, (?>\d+) ne peut retenir que la séquence entière de chiffres. Cette construction peut contenir un nombre arbitraire de sous-masques complexes, et ils peuvent être imbriqués.

Les sous-masques uniques ne peuvent être utilisés qu'avec les assertions arrières, pour effectuer une recherche efficace en fin de chaîne. Considérons un masque simple tel que

```
abcd$
```

appliqué à une très longue chaîne qui ne lui correspond pas. A cause du système de recherche de gauche à droite, PCRE va commencer par rechercher un "a" dans la chaîne sujet, puis vérifier si ce qui suit convient au reste du masque. Si le masque est spécifié sous la forme

```
^.*abcd$
```

alors, la séquence.* remplace en premier lieu la chaîne entière, et échoue, repart en arrière, et remplace tous les caractères sauf le dernier, échoue, retourne en arrière, prend un caractère de moins, etc... et ainsi de suite. Encore une fois, la recherche du "a" passe en revue toute la chaîne de gauche à droite, ce qui n'est pas très efficace. Par contre, si le masque était écrit

```
^(?>.*)(?<=abcd)
```

alors il n'y aurait pas de retour en arrière, pour satisfaire la séquence.*; car elle ne peut que remplacer toute la chaîne. L'assertion arrière consécutive va alors faire un test sur les 4 derniers caractères. Si elle échoue, la recherche est immédiatement interrompue. Pour les chaînes très longues, cette approche fait la différence en terme de performance et de temps de recherche. Lorsqu'un masque contient une répétition illimitée dans un sous-masque, qui contient lui-même un nombre illimité de répétiteur, l'utilisation des sous-masques à utilisation unique sont la seule façon d'éviter l'échec de la recherche à cause d'un temps de calcul trop long.

Le masque

```
(\D+|<\d+>)*[!?)
```

recherche un nombre illimité de sous-chaînes, qui contiennent soit des non-chiffres, ou alors des chiffres inclus dans

```
<>
```

, suivi soit par ! ou par ?. Lorsqu'il trouve une solution, ce masque va très vite. Mais, lorsqu'il est appliqué à une chaîne telle que :

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

Il lui faut beaucoup de temps pour annoncer un échec. Cela est dû au fait que la chaîne peut être divisée en deux sous-chaînes d'un grand nombre de façons, et qu'elles ont toutes été essayées.

(Cet exemple utilisait [!?] plutôt qu'un caractère simple, car PCRE et PHP utilise une optimisation qui permet de détecter rapidement l'échec lorsqu'un caractère unique est trouvé. Il se souvient du dernier caractère qui est attendu, et s'aperçoit rapidement qu'il n'y a pas ce caractère.). Si le masque utilisé est

```
((?>\d+)|lt;\d+>)*[!?]
```

les séquences de chiffres ne peuvent pas être trouvées, et l'échec intervient rapidement.

Les sous-masques conditionnels (CONDITIONAL SUBPATTERNS)

Il est possible de lier un sous-masque à une condition, ou de choisir entre deux sous-masques alternatifs, en fonction du résultat d'une assertion, ou suivant les résultats de recherche précédents. Les deux formes possibles de sous-masques conditionnels sont

```
(?(condition)masque positif)
```

```
(?(condition) masque positif | masque négatif)
```

Si les conditions sont satisfaites, le masque positif est utilisé, sinon, le masque négatif est utilisé, si présent. Si il y a plus de deux alternatives, une erreur est générée à la compilation.

Il y a deux types de conditions. Si le texte entre les parenthèses est une séquence de chiffre, alors la condition est satisfaite si le sous-masque correspondant à ce numéro a réussi. Considérons le masque suivant, qui contient des espaces non significatifs pour le rendre plus compréhensible (on supposera l'option PCRE_EXTENDED mise) et qui est divisé en trois parties pour simplifier les explications

```
( \ ( )? [ ^ ( ) ] + ( ? ( 1 ) \ ) )
```

La première partie recherche une parenthèse ouvrante optionnelle, et si elle existe, elle est capturée. La deuxième partie recherche une séquence de caractères qui ne contient pas de parenthèses. La troisième partie est conditionnée à la première, et s'assure que si il y avait une parenthèse ouvrante, il en existe une fermante. Si une parenthèse ouvrante a été trouvée, elle a été capturée, et donc la première capture existe, et la condition est exécutée. Sinon, elle est ignorée. Ce masque recherche donc une séquence de lettre, éventuellement mis entre parenthèse. Si la condition n'est pas une séquence de nombre, il faut que ce soit une assertion. Ce peut être une assertion positive ou négative, arrière ou avant. Considérons le masque suivant (même conditions que le précédent) et avec deux alternatives en seconde ligne :

```
(?([?=[^a-z]*[a-z])
\d{2}[a-z]{3}-\d{2} | \d{2}-\d{2}-\d{2} )
```

La condition est une assertion avant positive, qui recherche une séquence optionnelle de caractère non-lettre. En d'autres termes, elle teste la présence d'au moins une lettre dans la chaîne sujet. Si une lettre est trouvée, la recherche se poursuit avec la première alternative, et sinon, avec la seconde. Ce masque recherche des chaînes de la forme dd-aaa-dd ou dd-dd-dd, avec aaa qui sont des lettres, et dd qui sont des chiffres

COMMENTS

La séquence (?# marque le début des commentaires, qui se termine à la prochaine parenthèse fermante. Les parenthèses imbriquées ne sont pas autorisées. Les caractères entre ces délimiteurs ne jouent alors aucun rôle dans le masque.

Si l'option PCRE_EXTENDED est mise, les caractères dièses # non échappés en dehors d'une classe de caractères introduisent un commentaire qui continuera jusqu'à la prochaine ligne dans le masque.

Masques récursifs

Considérons le cas où il faut rechercher dans une chaîne, avec un niveau d'imbrication infini de parenthèses. Sans l'aide de la récursivité, le mieux que nous puissions obtenir est de créer un masque avec un niveau fixé de profondeur d'imbrication. Il n'est pas possible de traiter des masques à niveau d'imbrications variable. PCRE fournit un nouvel outil expérimental qui permet d'utiliser la récursivité dans les masques (entre autre). L'option (?R) est fournie pour servir la cause de la récursivité. Le masque suivant résout le problème des parenthèses (en utilisant l'option PCRE_EXTENDED est utilisée pour

ignorer les espaces) :

```
\( ( (?>[^( )]+) | (?R) )* \)
```

Tout d'abord, le masque recherche une parenthèse ouvrante. Puis, il recherche n'importe quel nombre de sous-chaînes qui sont soit des séquences de caractères non-parenthèses, ou bien une recherche récursive avec le même masque (i.e. une chaîne correctement inclus entre parenthèses). Finalement, il recherche une parenthèse fermante.

Cet exemple particulier contient un nombre illimité de répétitions imbriquées, ce qui fait que l'utilisation de sous-chaînes à utilisation unique pour rechercher les séquence de caractères non-parenthèses est important, lorsqu'il s'applique à une chaîne qui n'est pas valide. Par exemple, si on l'applique à

```
(aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa())
```

la réponse arrive rapidement. Sinon, si les sous-chaînes à utilisation unique ne sont pas utilisées, la recherche peut prendre un très long temps, car il existe de très nombreuses combinaisons de

+ et * à tester avant de conclure à l'échec.

Les valeurs utilisées pour capturer les sous-masques sont celles utilisées par les niveaux les plus hauts de récursivités, auquel la valeur est fixée. Si le masque précédent est utilisé avec

```
(ab(cd)ef)
```

la valeur de la parenthèse capturant est "ef", qui est la dernière valeur lue au niveau supérieur. Si de nouvelles parenthèses sont ajoutées, par exemple :

```
\( ( ( (?>[^( )]+) | (?R) )* ) \)
  ^               ^
  ^               ^
```

alors la chaîne capturée est "ab(cd)ef", c'est à dire le contenu de la parenthèses capturant de plus haut niveau. S'il y a plus de 15 parenthèses capturantes dans une chaîne, PCRE doit utiliser plus de mémoire pour stocker ces données. S'il ne peut obtenir cette mémoire supplémentaire, il ne fait que sauver les 15 premières, car il n'y a pas moyen de générer une erreur de mémoire lors d'une récursion.

PERFORMANCE

Certaines séquences de recherches sont plus efficaces que d'autres. Ainsi, il est plus efficace d'utiliser une classe de caractères telle que [aeiou] plutôt qu'une alternative (a|e|i|o|u).

En général, le masque le plus simple, qui permette la recherche désirée est généralement le plus efficace. Le livre de Jeffrey Friedl's contient de nombreuses études à propos de l'optimisation des expressions régulières.

Lorsqu'un masque commence par.* et que l'option PCRE_DOTALL est mise, le masque est implicitement ancré par PCRE, étant donné qu'il ne peut que rechercher au début de la chaîne. Cependant, si option PCRE_DOTALL n'est pas mise, PCRE ne peut faire aucune optimisation car le méta-caractères point (.). ne remplace pas une nouvelle ligne, et si la chaîne sujet contient des nouvelles lignes, le masque peut trouver une solution qui serait située juste après une de ces nouvelles lignes, et non pas seulement au début de la chaîne sujet. Par exemple, le masque, (.*) second

acceptera la chaîne "premier \n second" (avec \n qui remplace la nouvelle ligne), et la première chaîne capturée sera "et". Afin d'effectuer la recherche, PCRE va essayer d'appliquer le masque à partir de chaque début de ligne.

Si vous utilisez un tel masque avec des chaînes qui ne contiennent pas de caractères de nouvelles lignes, les meilleures performances seront atteintes avec l'option PCRE_DOTALL, ou en ancrant le masque avec ^. *. Cela évite à PCRE de scanner toute la chaîne pour rechercher un caractère de nouvelle ligne et recommencer la recherche.

Attention aux masques qui contiennent des quantificateurs infinis imbriqués. Ils peuvent demander un temps de calcul très long, lorsqu'appliqués à une chaîne qui n'accepte pas ce masque. Par exemple,

```
(a+)*
```

peut accepter "aaaa" de 33 manières différentes, et ce nombre

croît rapidement avec la taille de la chaîne (le quantificateur * peut prendre les valeurs de 0, 1, 2, 3, ou 4, et pour chaque cas non nul, le quantificateur + peut prendre différentes valeurs). Lorsque le reste de la chaîne est tel que l'on s'achemine vers un échec, PCRE doit en principe vérifier toutes les possibilités, et cela prend un temps extrêmement long.

Un optimiseur repère les cas les plus simples, tel que

```
(a+)*b
```

où un caractère simple suit les quantificateurs. Avant de partir dans les procédures standard de recherche, PCRE s'assure qu'il y a au moins un "b" dans la chaîne, et si ce n'est pas le cas, l'échec est annoncé immédiatement. Sinon, il n'y a pas d'optimisation dans la recherche. Vous pouvez voir la différence de comportement avec le masque suivant :

```
(a+)*\d
```

Le premier retourne un échec quasi immédiatement, s'il est appliqué à une ligne de "a", alors que le second masque prend un temps significatif pour une chaîne de plus de 20 caractères.

10.54 PDF

[\[Notes en ligne\]](#)

Vous disposez de fonctions PDF en PHP pour créer des fichiers PDF, pour peu que vous ayez la bibliothèque PDF de Thomas Merz (disponible à : <http://www.pdflib.com/pdflib/index.html> (site anglais)). Vous aurez aussi besoin des librairies [JPEG library](#), [the TIFF library](#), pour compiler cette librairie. Ces deux librairies posent pas mal de problèmes lors de la configuration. Suivez attentivement les messages d'erreur. Reportez vous à l'excellente documentation de pdflib, disponible avec la distribution de pdflib. C'est une introduction très pratique des capacités de pdflib. La plus part des fonctions de pdflib se retrouvent dans PHP sous le même nom. De même, les paramètres sont identiques. Vous devez connaître les concepts de base de PDF ou de Postscript pour utiliser efficacement ce module. Toutes les longueurs et coordonnées sont mesurées en points Postscript points. Il y a généralement 72 points PostScript par pouce, mais cela dépend en fait de la résolution d'affichage.

Il y a un autre module PHP pour créer des document PDF, basé sur la bibliothèque [FastIO's's ClibPDF](#). Les API sont légèrement différentes. Reportez vous à la section [10.9 ClibPDF](#) pour plus de détails.

Le module PDF introduit un nouveau type de variables. C'est *pdfdoc* : c'est un pointeur sur un document PDF et toutes les fonctions l'utilise comme premier paramètre.

10.54.1 Confusion entre les vieilles version de pdflib

[\[Notes en ligne\]](#)

Depuis le début du support de PDF sous PHP, (commençant avec la version pdflib 0.6), il y a eu des milliers de modifications dans les API de pdflib. La plus part de ces modifications ont été suivies par PHP, et parfois même au prix de modifications des API PHP. Depuis la version 3.x, ces API semblent s'être stabilisées, et PHP 4 a adoptée cette version comme le minimum nécessaire pour supporter PDF. En conséquence de quoi, un grand nombre de fonction vont disparaître, ou être remplacée. Le support de pdflib 0.6 est complètement abandonné. La liste suivante indique quelles sont les fonctions obsolète dans PHP 4.02, et qui devraient être remplacées par de nouvelles versions.

Anciennes fonctions	Nouvelles fonctions
pdf_put_image()	Désormais inutile
pdf_get_font()	pdf_get_value() avec "font" comme

	second paramètre.
<code>pdf_get_fontsize()</code>	pdf_get_value() avec "fontsize" comme second paramètre.
<code>pdf_get_fontname()</code>	pdf_get_parameter() avec "fontname" comme second paramètre.
<code>pdf_set_info_creator()</code>	pdf_set_info() avec "Creator" comme second paramètre.
<code>pdf_set_info_title</code>	pdf_set_info() avec "Title" comme second paramètre.
<code>pdf_set_info_subject()</code>	pdf_set_info() avec "Subject" comme second paramètre.
<code>pdf_set_info_author</code>	pdf_set_info() avec "Author" comme second paramètre.
<code>pdf_set_info_keywords</code>	pdf_set_info() avec "Keywords" comme second paramètre.
pdf_set_leading()	pdf_set_value() avec "leading" comme second paramètre.
pdf_set_text_rendering()	pdf_set_value() avec "textrendering" comme second paramètre.
pdf_set_text_rise()	pdf_set_value() avec "textrise" comme second paramètre.
pdf_set_horiz_scaling()	pdf_set_value() avec "horizscaling" comme second paramètre.
pdf_set_text_matrix()	Désormais inutile
pdf_set_char_spacing()	pdf_set_value() avec "charspacing" comme second paramètre.
pdf_set_word_spacing()	pdf_set_value() avec "wordspacing" comme second paramètre.
pdf_set_transition()	pdf_set_parameter() avec "transition" comme second paramètre.
pdf_set_duration()	pdf_set_value() avec "duration" comme second paramètre.
pdf_open_gif()	pdf_open_image_file() avec "gif" comme second paramètre.
pdf_open_jpeg()	pdf_open_image_file() avec "jpeg" comme second paramètre.
pdf_open_tiff()	pdf_open_image_file() avec "tiff" comme second paramètre.
pdf_open_png()	pdf_open_image_file() avec "png" comme second paramètre.
<code>pdf_get_imagewidth</code>	pdf_get_value() avec "imagewidth"

	comme second paramètre et l'image comme troisième.
pdf_get_imageheight	pdf_get_value() avec "imageheight" comme second paramètre et l'image comme troisième.

10.54.2 Conseils pour installer pdflib 3.x

[\[Notes en ligne\]](#)

Depuis la version 3.0 de pdflib vous pouvez configurer cette librairie avec l'option `--enable-shared-pdflib`.

10.54.3 Installation des anciennes versions de pdflib

[\[Notes en ligne\]](#)

Si vous utilisez pdflib 2.01 vérifiez comment votre librairie a été installée. Il doit y avoir un fichier (ou un lien) vers libpdf.so. La version 2.01 ne fait que créer une librairie avec le nom libpdf2.01.so qui ne peut être trouvé lors de la compilation du programme de configuration. Vous devez créer vous même ce lien symbolique de libpdf.so vers libpdf2.01.so..

La version 2.20 de pdflib a introduit de nombreuses modifications dans ses API, ainsi que le support des polices chinoises et japonaises. Cela impliquent malheureusement des modifications dans le module PDF de PHP 4 (mais pas de PHP 3). Si vous utilisez pdflib 2.20, gérer correctement votre mémoire. Jusqu'à la version 3.0, pdflib peut se révéler très instable. Le paramètre d'encodage [pdf_set_font\(\)](#) est devenu une chaîne. Cela signifie notamment qu'il faut remplacer 4 par 'winansi'.

Si vous utilisez pdflib 2.30, [pdf_set_text_matrix\(\)](#) a disparu. Elle n'est plus supporté. En général, il est recommandé de consulter les notes de version de la pdflib pour lister toutes les modifications.

A partir du 9 mars 2000, PHP 4 ne supporte plus que la version 3.0 et plus récente de pdflib. PHP 3, par contre, ne doit pas être utilisé avec des versions plus récentes que la 2.01.

10.54.4 Exemples

[\[Notes en ligne\]](#)

La plus part des fonctions sont simples d'emploi. Le plus difficile est probablement de créer un fichier PDF simple. L'exemple suivant devrait vous mettre sur les rails. Il crée un fichier `test.pdf` d'une page. La page contient du texte "Times Roman outlined", de taille de 30pt. Le texte est aussi souligné.

Création d'un document PDF avec pdflib

```
<?php
$fp = fopen("test.pdf", "w");
$pdf = pdf_open($fp);
pdf_set_info($pdf, "Author", "Uwe Steinmann");
pdf_set_info($pdf, "Title", "Test for PHP wrapper of PDFlib 2.0");
pdf_set_info($pdf, "Creator", "See Author");
pdf_set_info($pdf, "Subject", "Testing");
pdf_begin_page($pdf, 595, 842);
pdf_add_outline($pdf, "Page 1");
pdf_set_font($pdf, "Times-Roman", 30, "host");
pdf_set_value($pdf, "textrendering", 1);
```

```
pdf_show_xy($pdf, "Times Roman outlined", 50, 750);
pdf_moveto($pdf, 50, 740);
pdf_lineto($pdf, 330, 740);
pdf_stroke($pdf);
pdf_end_page($pdf);
pdf_close($pdf);
fclose($fp);
echo "<A HREF=getpdf.php>finished</A>";
?>
```

Ce script ``getpdf.php`` retourne simplement le document PDF.

```
<?php
$fp = fopen("test.pdf", "r");
header("Content-type: application/pdf");
fpassthru($fp);
fclose($fp);
?>
```

La distribution `pdflib` contient un exemple plus complexe, qui crée des pages plus élaborées, avec une horloge. Cet exemple a été converti en script PHP (vous retrouverez cet exemple dans le module [10.9 ClibPDF](#)) :

Exemple `pdfclock` issue de la distribution `pdflib`

```
<?php
$pdffilename = "clock.pdf";
$radius = 200;
$margin = 20;
$pagecount = 40;
$fp = fopen($pdffilename, "w");
$pdf = pdf_open($fp);
pdf_set_info($pdf, "Creator", "pdf_clock.php3");
pdf_set_info($pdf, "Author", "Uwe Steinmann");
pdf_set_info($pdf, "Title", "Analog Clock");
while($pagecount-- > 0) {
    pdf_begin_page($pdf, 2 * ($radius + $margin), 2 * ($radius + $margin));
    pdf_set_parameter($pdf, "transition", "wipe");
    pdf_set_value($pdf, "duration", 0.5);
    pdf_translate($pdf, $radius + $margin, $radius + $margin);
    pdf_save($pdf);
    pdf_setrgbcolor($pdf, 0.0, 0.0, 1.0);
    /* minute strokes */
    pdf_setlinewidth($pdf, 2.0);
    for ($alpha = 0; $alpha < 360; $alpha += 6) {
        pdf_rotate($pdf, 6.0);
        pdf_moveto($pdf, $radius, 0.0);
        pdf_lineto($pdf, $radius-$margin/3, 0.0);
        pdf_stroke($pdf);
    }
    pdf_restore($pdf);
    pdf_save($pdf);
    /* 5 minute strokes */
    pdf_setlinewidth($pdf, 3.0);
    for ($alpha = 0; $alpha < 360; $alpha += 30) {
        pdf_rotate($pdf, 30.0);
        pdf_moveto($pdf, $radius, 0.0);
        pdf_lineto($pdf, $radius-$margin, 0.0);
        pdf_stroke($pdf);
    }
}
```

```

    }
    $ltime = getdate();
    /* draw hour hand */
pdf_save($pdf);
pdf_rotate($pdf, -((($ltime['minutes']/60.0)+$ltime['hours']-3.0)*30.0);
pdf_moveto($pdf, -$radius/10, -$radius/20);
pdf_lineto($pdf, $radius/2, 0.0);
pdf_lineto($pdf, -$radius/10, $radius/20);
pdf_closepath($pdf);
pdf_fill($pdf);
pdf_restore($pdf);
    /* draw minute hand */
pdf_save($pdf);
pdf_rotate($pdf, -((($ltime['seconds']/60.0)+$ltime['minutes']-15.0)*6.0);
pdf_moveto($pdf, -$radius/10, -$radius/20);
pdf_lineto($pdf, $radius * 0.8, 0.0);
pdf_lineto($pdf, -$radius/10, $radius/20);
pdf_closepath($pdf);
pdf_fill($pdf);
pdf_restore($pdf);
    /* draw second hand */
pdf_setrgbcolor($pdf, 1.0, 0.0, 0.0);
pdf_setlinewidth($pdf, 2);
pdf_save($pdf);
pdf_rotate($pdf, -((($ltime['seconds'] - 15.0) * 6.0));
pdf_moveto($pdf, -$radius/5, 0.0);
pdf_lineto($pdf, $radius, 0.0);
pdf_stroke($pdf);
pdf_restore($pdf);
    /* draw little circle at center */
pdf_circle($pdf, 0, 0, $radius/30);
pdf_fill($pdf);
pdf_restore($pdf);
pdf_end_page($pdf);
}
$pdf = pdf_close($pdf);
fclose($fp);
echo "<A HREF=getpdf.php?filename=".$pdffilename.">finished</A>";
?>

```

Ce script ``getpdf.php`` ne fait que retourner un document PDF.

```

<?php
$fp = fopen($filename, "r");
header("Content-type: application/pdf");
fpassthru($fp);
fclose($fp);
?>

```

10.54.5 pdf_set_info

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_info](#)(int *pdf document*, string *fieldname*, string *value*)
 [PHP 4 >= 4.0.1]

[pdf_set_info\(\)](#) modifie un champs d'entête d'un document PDF. Les valeurs possibles pour *fieldname* sont :

'Subject' (sujet), 'Title' (titre), 'Creator' (créateur), 'Author' (auteur), 'Keywords' (mots-clé) et un autre nom, défini par l'utilisateur. Cette fonction peut être appelée avant la création d'une page.

Préparer l'entête d'un document PDF

```
<?php
$fd = fopen("test.pdf", "w");
$pdfdoc = pdf_open($fd);
pdf_set_info($pdfdoc, "Author", "Uwe Steinmann");
pdf_set_info($pdfdoc, "Creator", "Uwe Steinmann");
pdf_set_info($pdfdoc, "Title", "Testing Info Fields");
pdf_set_info($pdfdoc, "Subject", "Test");
pdf_set_info($pdfdoc, "Keywords", "Test, Fields");
pdf_set_info($pdfdoc, "CustomField", "What ever makes sense");
pdf_begin_page($pdfdoc, 595, 842);
pdf_end_page($pdfdoc);
pdf_close($pdfdoc);
?>
```

Note : Cette fonction remplace `pdf_set_info_keywords`, `pdf_set_info_title`, `pdf_set_info_subject`, `pdf_set_info_creator`, `pdf_set_info_subject`.

10.54.6 pdf_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open](#) (int *file*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_open\(\)](#) ouvre un nouveau document PDF. Le fichier correspondant doit avoir été ouvert avec [fopen\(\)](#) et le pointeur de fichier est passé en argument *file*. Si vous ne passez aucun paramètres, le document sera créé en mémoire, et envoyé à la sortie standard (stdout ou client web).

Note : La valeur retournée est utilisée par toutes les autres fonctions PDF comme premier argument.

Voir aussi [fopen\(\)](#), [pdf_close\(\)](#).

10.54.7 pdf_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_close](#) (int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_close\(\)](#) ferme un document PDF.

Voir aussi [pdf_open\(\)](#), [fclose\(\)](#).

10.54.8 pdf_begin_page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_begin_page](#) (int *pdf document*, double *width*, double *height*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_begin_page\(\)](#) commence une nouvelle page avec la taille *height* et la largeur *width*. Afin de créer un nouveau document valide, vous devez appeler cette fonction et [pdf_end_page\(\)](#) au moins une fois.

Voir aussi [pdf_end_page\(\)](#).

10.54.9 pdf_end_page

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_end_page](#)(int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_end_page\(\)](#) termine une page. Une fois qu'une page a été fermée, elle ne peut pas être modifiée.

Voir aussi [pdf_begin_page\(\)](#).

10.54.10 pdf_show

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_show](#)(int *pdf document*, string *text*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_show\(\)](#) affiche le texte *text* avec la position courante, et avec la police courante.

Voir aussi [pdf_show_xy\(\)](#), [pdf_show_boxed\(\)](#), [pdf_set_text_pos\(\)](#), [pdf_set_font\(\)](#).

10.54.11 pdf_show_boxed

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_show_boxed](#)(int *pdf document*, string *text*, double *x-coor*, double *y-coor*, double *width*, double *height*, string *mode*, string *feature*)

[PHP 4 >= 4.0RC1]

[pdf_show_boxed\(\)](#) affiche le texte *text* dans un rectangle, dont le coin inférieur gauche est aux coordonnées (*x-coor*, *y-coor*). Les dimensions du rectangle sont *height* et *width*. Le paramètre *mode* indique le type de *text*. Si *width* et *height* sont à zéro, le mode *mode* peut être "left" (gauche), "right" (droite) ou "center" (centré). *width* ou *height* sont différents pouvant prendre les valeurs de "justify" (justification) ou "fulljustify" (justification complète).

Si le paramètre *feature* vaut "blind", le texte n'est pas affiché.

Retourne le nombre de caractères qui n'ont pas pu être traités, car ils ne rentraient pas dans le rectangle.

Voir aussi [pdf_show\(\)](#) et [pdf_show_xy\(\)](#).

10.54.12 pdf_show_xy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_show_xy](#)(int *pdf document*, string *text*, double *x-coor*, double *y-coor*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_show_xy\(\)](#) affiche le texte *text* à la position donnée par les coordonnées (*x-coor*, *y-coor*).

Voir aussi [pdf_show\(\)](#), [pdf_show_boxed\(\)](#).

10.54.13 pdf_set_font

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_font](#)(int *pdf document*, string *font name*, double *size*, string *encoding*, int *embed*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_set_font\(\)](#) sélectionne la police, sa taille et son encodage. Il vous faudra fournir des fichiers Adobe Font Metrics (afm) comme police, dans le dossier de police (par défaut ./fonts). Si vous utilisez pdflib 0.6, vous devrez fournir des fichiers Adobe Font Métrique (afm-files) pour les polices, dans le chemin de police (par défaut, ./fonts). Si vous utilisez php versin 3 ou une version plus ancienne que la version 2.20 de pdflib, le quatrième paramètre *encoding* peut prendre les valeurs suivantes : 0 = builtin, 1 = pdfdoc, 2 = macroman, 3 = macexpert, 4 = winansi. Un encodage plus grand que 4 et inférieur à 0 sera transformé en 'winansi'. 'winansi' est souvent un bon choix. Si vous utilisez PHP version 4 et une version plus ancienne que la version 2.20 de pdflib le quatrième paramètre *encoding* est une chaîne : 'builtin', 'pdfdoc', 'macroman', 'macexpert', 'winansi'. Si le dernier paramètre est à 1, la police est intégrée dans le document. Sinon, elle ne le sera pas. Incorporer une police dans un document est une bonne idée si la police n'est pas répandue, ou si vous ne pouvez pas vous assurer que le la personne qui regardera votre document peut accéder à cette police.

Note : [pdf_set_font\(\)](#) doit être appelée après [pdf_begin_page\(\)](#) pour créer un document PDF valide.

Note : Si vous référencez une police dans un fichier .upr, assurez vous que le nom du fichier .afm et celui de la police sont bien les mêmes. Sinon, la police sera agrandie plusieurs fois (Merci à Paul Haddon pour cette info).

10.54.14 pdf_set_leading

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_leading](#)(int *pdf document*, double *distance*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_set_leading\(\)](#) permet de choisir la distance entre les lignes du texte. Cette valeur sera utilisée si du texte est affiché par [pdf_continue_text\(\)](#).

Voir aussi [pdf_continue_text\(\)](#).

10.54.15 pdf_set_parameter

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_parameter](#)(int *pdf document*, string *name*, string *value*)

[PHP 4 >= 4.0RC1]

[pdf_set_parameter\(\)](#) modifie certaines valeurs de pdglib. *value* est de type chaîne.

Voir aussi [pdf_get_value\(\)](#), [pdf_set_value\(\)](#), [pdf_get_parameter\(\)](#).

10.54.16 pdf_get_parameter

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pdf_get_parameter](#)(int *pdf document*, string *name*, double *modifier*)

[PHP 4 >= 4.0.1]

[pdf_get_parameter\(\)](#) lit certains paramètres de la librairie pdflib. Ces paramètres sont des chaînes de caractères. Le paramètre *modifier* caractérise le paramètre. S'il n'est pas nécessaire, il faut l'omettre, ou passer 0.

Voir aussi [pdf_get_value\(\)](#), [pdf_set_value\(\)](#), [pdf_set_parameter\(\)](#).

10.54.17 pdf_set_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_value](#)(int *pdf document*, string *name*, double *value*)
[PHP 4 >= 4.0.1]

[pdf_set_value\(\)](#) modifie la valeur (numérique) du paramètre *name* de PDFlib.

Voir aussi [pdf_get_value\(\)](#), [pdf_get_parameter\(\)](#), [pdf_set_parameter\(\)](#).

10.54.18 pdf_get_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [pdf_get_value](#)(int *pdf document*, string *name*, double *modifier*)
[PHP 4 >= 4.0.1]

[pdf_get_parameter\(\)](#) lit certains paramètres de la librairie pdflib. Ces paramètres sont des numériques. Le paramètre *modifier* caractérise le paramètre. S'il n'est pas nécessaire, il faut l'omettre, ou passer 0.

Voir aussi [pdf_get_parameter\(\)](#), [pdf_set_value\(\)](#), [pdf_set_parameter\(\)](#).

10.54.19 pdf_get_image_height

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pdf_get_image_height](#)(int *pdf document*, int *image*)
[PHP 3>= 3.0.12, PHP 4 >= 4.0b2]

[pdf_get_image_height\(\)](#) retourne la hauteur de l'image *image* en pixels.

Voir aussi [pdf_open_image_file\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_get_image_width\(\)](#).

10.54.20 pdf_get_image_width

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pdf_get_image_width](#)(int *pdf document*, int *image*)
[PHP 3>= 3.0.12, PHP 4 >= 4.0b2]

[pdf_get_image_width\(\)](#) retourne la largeur de l'image *image*, en pixels.

Voir aussi [pdf_open_image_file\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_get_image_height\(\)](#).

10.54.21 pdf_set_text_rendering

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_rendering](#)(int *pdf document*, int *mode*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_set_text_rendering\(\)](#) détermine le rendu du texte. Les valeurs possibles pour *mode* sont 0=fill text (texte plein), 1=stroke text (???), 2=fill et stroke text (texte plein et stroke), 3=invisible, 4=texte plein, et ajouté au chemin, 5=stroke text, ajouté au chemin, 6=texte plein et stroke, ajouté au chemin, 7=ajouté au chemin.

10.54.22 pdf_set_horiz_scaling

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_horiz_scaling](#) (int *pdf document*, double *scale*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_set_horiz_scaling\(\)](#) fixe l'échelle horizontale du texte, à *scale* en pourcentage.

10.54.23 pdf_set_text_rise

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_rise](#) (int *pdf document*, double *rise*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_set_text_rise\(\)](#) fixe l'élévation du texte à *rise* points.

10.54.24 pdf_set_text_matrix

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_matrix](#) (int *pdf document*, array *matrix*)
[PHP 3>= 3.0.6, PHP 4 <= 4.0b4]

[pdf_set_text_matrix\(\)](#) choisit la matrice de texte : la matrice de texte détermine les transformations de la police courante. La matrice est un tableau de 6 éléments.

10.54.25 pdf_set_text_pos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_text_pos](#) (int *pdf document*, double *x-coor*, double *y-coor*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_set_text_pos\(\)](#) modifie la position du texte qui sera utilisée lors du prochain [pdf_show\(\)](#). Voir aussi [pdf_show\(\)](#), [pdf_show_xy\(\)](#).

10.54.26 pdf_set_char_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_char_spacing](#) (int *pdf document*, double *space*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_set_char_spacing\(\)](#) fixe l'espacement des caractères. Voir aussi [pdf_set_word_spacing\(\)](#), [pdf_set_leading\(\)](#).

10.54.27 pdf_set_word_spacing

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_word_spacing](#)(int *pdf document*, double *space*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_set_word_spacing\(\)](#) modifie l'espacement des mots.

Voir aussi [pdf_set_char_spacing\(\)](#), [pdf_set_leading\(\)](#).

10.54.28 pdf_skew

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_skew](#)(int *pdf document*, double *alpha*, double *beta*)

[PHP 4 >= 4.0RC1]

[pdf_skew\(\)](#) modifie le système de coordonnées, en faisant une rotation d'angle *alpha* pour les (x) et d'angle *beta* pour les (y), en degrés. *alpha* et *beta* ne peuvent pas prendre les valeurs de 90 ou 270 degrés.

10.54.29 pdf_continue_text

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_continue_text](#)(int *pdf document*, string *text*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_continue_text\(\)](#) affiche le texte *text* sur une nouvelle ligne. La distance entre les lignes peut être choisie avec [pdf_set_leading\(\)](#).

Voir aussi [pdf_show_xy\(\)](#), [pdf_set_leading\(\)](#), [pdf_set_text_pos\(\)](#).

10.54.30 pdf_stringwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [pdf_stringwidth](#)(int *pdf document*, string *text*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_stringwidth\(\)](#) retourne la largeur du texte *text* avec la police courante. Il faut qu'une police ait été choisie auparavant.

Voir aussi [pdf_set_font\(\)](#).

10.54.31 pdf_save

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_save](#)(int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_save\(\)](#) enregistre l'environnement courant. Le fonctionnement est identique à la commande postscript gsave. Très pratique si vous voulez faire une translation ou une rotation d'un objet, sans affecter les autres.

[pdf_save\(\)](#) sera toujours suivi d'un [pdf_restore\(\)](#).

Voir aussi [pdf_restore\(\)](#).

[10.54.32 pdf_restore](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_restore](#) (int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_restore\(\)](#) restaure un environnement sauvé par [pdf_save\(\)](#). Cela fonctionne de manière identique à la commande Postscript grestore. Très pratique lorsque vous vous faire des translations ou des rotations sans affecter les autres objets.

Sauver et restaurer un environnement PDF

```
<?php
pdf_save($pdf);
// tout un lot de rotations, translations, transformations...
pdf_restore($pdf)
?>
```

Voir aussi [pdf_save\(\)](#).

[10.54.33 pdf_translate](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_translate](#) (int *pdf document*, double *x-coor*, double *y-coor*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_translate\(\)](#) place l'origine du système de coordonnées au point (*x-coor*, *y-coor*). L'exemple suivant trace une ligne de (0, 0) à (200, 200) par rapport aux coordonnées initiales. Il faut aussi désigner le point courant après [pdf_translate\(\)](#) et avant de commencer à dessiner les objets.

Translation

```
<?php PDF_moveto($pdf, 0, 0);
PDF_lineto($pdf, 100, 100);
PDF_stroke($pdf);
PDF_translate($pdf, 100, 100);
PDF_moveto($pdf, 0, 0);
PDF_lineto($pdf, 100, 100);
PDF_stroke($pdf);
?>
```

[10.54.34 pdf_scale](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_scale](#) (int *pdf document*, double *x-scale*, double *y-scale*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_scale\(\)](#) modifie l'échelle dans les deux directions. L'exemple suivant multiplie l'échelle par 72. La ligne

suivante sera dessinée sur un pouce (2.54 cm) de large.

Mise à l'échelle

```
<?php PDF_scale($pdf, 72.0, 72.0);
PDF_lineto($pdf, 1, 1);
PDF_stroke($pdf);
?>
```

10.54.35 pdf_rotate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_rotate](#) (int *pdf document*, double *angle*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_rotate\(\)](#) modifie la rotation de *angle* degré.

10.54.36 pdf_setflat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setflat](#) (int *pdf document*, double *value*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_setflat\(\)](#) modifie la platitude, et lui affecte la valeur *value* entre 0 et 100.

10.54.37 pdf_setlinejoin

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setlinejoin](#) (int *pdf document*, long *value*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_setlinejoin\(\)](#) modifie le paramètre "linejoin", et lui affecte la valeur *value*, entre 0 et 2.

10.54.38 pdf_setlinecap

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setlinecap](#) (int *pdf document*, int *value*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_setlinecap\(\)](#) affecte au paramètre "linecap" la valeur *value*, entre 0 et 2.

10.54.39 pdf_setmiterlimit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setmiterlimit](#) (int *pdf document*, double *value*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_setmiterlimit\(\)](#) modifie la "miter limit" et lui affecte la valeur *value*, supérieure à 1.

[10.54.40 pdf_setlinewidth](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setlinewidth](#) (int *pdf document*, double *width*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_setlinewidth\(\)](#) affecte à largeur de ligne la valeur *width*.

[10.54.41 pdf_setdash](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setdash](#) (int *pdf document*, double *white*, double *black*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_setdash\(\)](#) modifie les caractères de remplissage, et affecte *white* comme caractère invisible, et *black* comme caractère de remplissage. Si les deux sont à zéros, une ligne continue est affichée.

[10.54.42 pdf_moveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_moveto](#) (int *pdf document*, double *x-coor*, double *y-coor*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_moveto\(\)](#) déplace le point courant à la position (*x-coor*, *y-coor*).

[10.54.43 pdf_curveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_curveto](#) (int *pdf document*, double *x1*, double *y1*, double *x2*, double *y2*, double *x3*, double *y3*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_curveto\(\)](#) dessine une courbe de Bezier entre le point courant et le point (*x3*, *y3*) en utilisant les points de contrôle (*x1*, *y1*) et (*x2*, *y2*).

Voir aussi [pdf_moveto\(\)](#), [pdf_lineto\(\)](#), [pdf_stroke\(\)](#).

[10.54.44 pdf_lineto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_lineto](#) (int *pdf document*, double *x-coor*, double *y-coor*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_lineto\(\)](#) dessine une ligne entre le point courant et le point de coordonnées (*x-coor*, *y-coor*).

Voir aussi [pdf_moveto\(\)](#), [pdf_curveto\(\)](#), [pdf_stroke\(\)](#).

[10.54.45 pdf_circle](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_circle](#)(int *pdf document*, double *x-coor*, double *y-coor*, double *radius*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_circle\(\)](#) dessine un cercle de centre (*x-coor*, *y-coor*), et de rayon *radius*.
Voir aussi [pdf_arc\(\)](#), [pdf_stroke\(\)](#).

[10.54.46 pdf_arc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_arc](#)(int *pdf document*, double *x-coor*, double *y-coor*, double *radius*,
double *start*, double *end*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_arc\(\)](#) dessine un arc de cercle, de centre (*x-coor*, *y-coor*) et de rayon *radius*, en commençant à l'angle *start* et finissant à l'angle *end*.
Voir aussi [pdf_circle\(\)](#), [pdf_stroke\(\)](#).

[10.54.47 pdf_rect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_rect](#)(int *pdf document*, double *x-coor*, double *y-coor*, double *width*,
double *height*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_rect\(\)](#) dessine un rectangle un rectangle de coin inférieur gauche de coordonnées (*x-coor*, *y-coor*). Sa longueur vaut *width*. Et sa largeur *height*.
Voir aussi [pdf_stroke\(\)](#).

[10.54.48 pdf_closepath](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_closepath](#)(int *pdf document*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_closepath\(\)](#) ferme et clos le chemin courant. Cela signifie qu'une ligne va être ajoutée entre le point courant et le premier point du chemin. De nombreuses fonctions telles que [pdf_moveto\(\)](#), [pdf_circle\(\)](#) et [pdf_rect\(\)](#) démarre un nouveau chemin.

[10.54.49 pdf_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_stroke](#)(int *pdf document*)
[PHP 3>= 3.0.6, PHP 4]

[pdf_stroke\(\)](#) dessine une ligne le long du chemin. Le chemin courant est la somme de toutes les lignes dessinées. Sans cette fonction, la ligne de chemin ne sera pas dessinée.

Voir aussi [pdf_closepath\(\)](#), [pdf_closepath_stroke\(\)](#).

10.54.50 pdf_closepath_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_closepath_stroke](#) (int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_closepath_stroke\(\)](#) est une combinaison de [pdf_closepath\(\)](#) et [pdf_stroke\(\)](#). Elle ferme aussi le chemin.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#).

10.54.51 pdf_fill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_fill](#) (int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_fill\(\)](#) remplit l'intérieur du chemin courant avec la couleur courante.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#), [pdf_setgray_fill\(\)](#), [pdf_setgray\(\)](#), [pdf_setrgbcolor_fill\(\)](#), [pdf_setrgbcolor\(\)](#).

10.54.52 pdf_fill_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_fill_stroke](#) (int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_fill_stroke\(\)](#) remplit l'intérieur du chemin courant avec la couleur courante, puis dessine le chemin courant.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#), [pdf_fill\(\)](#), [pdf_setgray_fill\(\)](#), [pdf_setgray\(\)](#), [pdf_setrgbcolor_fill\(\)](#), [pdf_setrgbcolor\(\)](#).

10.54.53 pdf_closepath_fill_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_closepath_fill_stroke](#) (int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_closepath_fill_stroke\(\)](#) clos le chemin, le remplit avec la couleur courante, et dessine le chemin.

Voir aussi [pdf_closepath\(\)](#), [pdf_stroke\(\)](#), [pdf_fill\(\)](#), [pdf_setgray_fill\(\)](#), [pdf_setgray\(\)](#), [pdf_setrgbcolor_fill\(\)](#), [pdf_setrgbcolor\(\)](#).

10.54.54 pdf_endpath

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_endpath](#)(int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_endpath\(\)](#) ferme le chemin courant mais ne le clos pas.

Voir aussi [pdf_closepath\(\)](#).

10.54.55 pdf_clip

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_clip](#)(int *pdf document*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_clip\(\)](#) aligne tous les dessins sur le chemin courant.

10.54.56 pdf_setgray_fill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setgray_fill](#)(int *pdf document*, double *gray value*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_setgray_fill\(\)](#) choisi la couleur grise comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_fill\(\)](#).

10.54.57 pdf_setgray_stroke

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setgray_stroke](#)(int *pdf document*, double *gray value*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_setgray_stroke\(\)](#) Fixe la couleur de dessin à un niveau de gris.

Voir aussi [pdf_setrgbcolor_stroke\(\)](#).

10.54.58 pdf_setgray

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setgray](#)(int *pdf document*, double *gray value*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_setgray\(\)](#) choisi la couleur grise comme couleur de remplissage et de dessin.

Voir aussi [pdf_setrgbcolor_stroke\(\)](#), [pdf_setrgbcolor_fill\(\)](#).

[10.54.59 pdf_setrgbcolor_fill](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setrgbcolor_fill](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_setrgbcolor_fill\(\)](#) choisi la couleur rgb comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_fill\(\)](#).

[10.54.60 pdf_setrgbcolor_stroke](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setrgbcolor_stroke](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_setrgbcolor_stroke\(\)](#) choisi la couleur rgb comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_stroke\(\)](#).

[10.54.61 pdf_setrgbcolor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_setrgbcolor](#) (int *pdf document*, double *red value*, double *green value*, double *blue value*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_setrgbcolor_stroke\(\)](#) choisi la couleur rgb comme couleur de remplissage.

Voir aussi [pdf_setrgbcolor_stroke\(\)](#), [pdf_setrgbcolor_fill\(\)](#).

[10.54.62 pdf_add_outline](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_add_outline](#) (int *pdf document*, string *text*, int *parent*, int *open*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_add_outline\(\)](#) ajoute un signet de nom *text* sur la page courante, et au point courant. Le signet est inséré comme un fils de la page *parent* et est ouvert par défaut si *open* n'est pas 0. La valeur retournée est un identifiant du signet, qui pourra être réutilisé comme parent d'autres signets. Vous pouvez ainsi créer des hiérarchies de signets.

Malheureusement, pdflib ne copie pas la chaîne, ce qui force PHP à allouer la mémoire. Actuellement, cette mémoire n'est jamais libérée par une fonction PDF, mais prise en charge par le gestionnaire PHP.

[10.54.63 pdf_set_transition](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_transition](#) (int *pdf document*, int *transition*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_set_transition\(\)](#) Fixe le mode de transition entre les pages. La valeur de *transition* peut être :

- 0 : aucune transition,
- 1 : deux lignes en travers de l'écran représente la page,
- 2 : plusieurs lignes en travers de l'écran représente la page,
- 3 : une boîte représente la page,
- 4 : une seule ligne en travers de l'écran représente la page,
- 5 : l'ancienne page se dissout pour révéler la nouvelle,
- 6 : l'effet page d'un coin à l'autre
- 7 : l'ancienne page est simplement remplacée par la nouvelle (valeur par défaut)

Voir aussi [pdf_set_duration\(\)](#).

[10.54.64 pdf_set_duration](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_duration](#) (int *pdf document*, double *duration*)

[PHP 3>= 3.0.6, PHP 4]

[pdf_set_duration\(\)](#) choisi la durée de transition, en secondes, entre deux pages.

Voir aussi [pdf_set_transition\(\)](#).

[10.54.65 pdf_open_gif](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_gif](#) (int *pdf document*, string *filename*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[pdf_open_gif\(\)](#) ouvre et charge l'image GIF du fichier *filename*. Le format doit être GIF. La fonction retourne un identifiant d'image PDF :

Inclusion d'un image GIF

```
<?php
$im = PDF_open_gif($pdf, "test.gif");
pdf_place_image($pdf, $im, 100, 100, 1);
pdf_close_image($pdf, $im);
?>
```

Voir aussi [pdf_close_image\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

[10.54.66 pdf_open_png](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_png](#) (int *pdf* , string *png_file*)

[PHP 4 >= 4.0RC2]

[pdf_open_png\(\)](#) ouvre une image, stockée dans le fichier *png_file*. Le format de cette image doit être PNG. La fonction retourne un identifiant d'image PNG. Note : *Cette fonction ne doit plus être utilisée. Utilisez plutôt [pdf_open_image_file\(\)](#).*

Including a PNG image

```
<?php
$im = pdf_open_png ($pdf, "test.png");
pdf_place_image ($pdf, $im, 100, 100, 1);
pdf_close_image ($pdf, $im);
?>
```

Voir aussi [pdf_close_image\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.54.67 pdf_open_image_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_image_file](#) (int *PDF-document*, string *format*, string *filename*)

[PHP 4 >= 4.0RC2]

[pdf_open_image_file\(\)](#) lit une image au format *format* dans le fichier *filename*. Les formats possibles sont 'png', 'tiff', 'jpeg' et 'gif'. La fonction retourne un identifiant d'image PDF.

Insertion d'une image dans un fichier PDF

```
<?php
$pim = pdf_open_image_file($pdf, "png", "picture.png");
pdf_place_image($pdf, $pim, 100, 100, 1);
pdf_close_image($pdf, $pim);
?>
```

Voir aussi [pdf_close_image\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_tiff\(\)](#), [pdf_open_png\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.54.68 pdf_open_memory_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_memory_image](#) (int *pdf document*, int *image*)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b2]

[pdf_open_memory_image\(\)](#) prend comme argument une image créée avec les fonctions PHP, et la rend disponible pour le document PDF. La fonction retourne un identifiant PDF d'image.

Inclusion d'une image mémoire

```
<?php
$im = ImageCreate(100, 100);
```

```

$col = ImageColorAllocate($im, 80, 45, 190);
ImageFill($im, 10, 10, $col);
$pim = PDF_open_memory_image($pdf, $im);
ImageDestroy($im);
pdf_place_image($pdf, $pim, 100, 100, 1);
pdf_close_image($pdf, $pim);
?>

```

Voir aussi [pdf_close_image\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_gif\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.54.69 pdf_open_jpeg

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_jpeg](#)(int *pdf document*, string *filename*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[pdf_open_jpeg\(\)](#) ouvre et charge l'image JPEG du fichier *filename*. Le format de l'image doit être JPEG. La fonction retourne un identifiant d'image PDF. Note : *Cette fonction ne doit plus être utilisée. Utilisez plutôt pdf_open_image_file().*

Voir aussi [pdf_close_image\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.54.70 pdf_open_tiff

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pdf_open_tiff](#)(int *PDF-document*, string *filename*)

[PHP 4 >= 4.0b4]

[pdf_open_jpeg\(\)](#) ouvre et charge l'image JPEG du fichier *filename*. Le format de l'image doit être JPEG. La fonction retourne un identifiant d'image PDF.

Note : *Cette fonction ne doit plus être utilisée. Utilisez plutôt pdf_open_image_file().*

Voir aussi [pdf_close_image\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_jpeg\(\)](#), [pdf_open_png\(\)](#), [pdf_open_memory_image\(\)](#), [pdf_execute_image\(\)](#), [pdf_place_image\(\)](#), [pdf_put_image\(\)](#).

10.54.71 pdf_close_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_close_image](#)(int *image*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[pdf_close_image\(\)](#) ferme une image qui a été ouverte par [pdf_open_gif\(\)](#) ou [pdf_open_jpeg\(\)](#).

Voir aussi [pdf_open_jpeg\(\)](#), [pdf_open_gif\(\)](#), [pdf_open_tiff\(\)](#), [pdf_open_memory_image\(\)](#).

10.54.72 pdf_place_image

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_place_image](#)(int *pdf document*, int *image*, double *x-coor*, double *y-coor*, double *scale*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[pdf_place_image\(\)](#) place l'image *image* dans la page courante, à la position (*x-coor*, *y-coor*). L'image peut changer d'échelle simultanément.

Voir aussi [pdf_put_image\(\)](#).

10.54.73 [pdf_put_image](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_put_image](#)(int *pdf document*, int *image*)

[PHP 3>= 3.0.7, 4.0b2 – 4.0b4 only]

[pdf_put_image\(\)](#) enregistre une image dans un fichier PDF pour utilisation ultérieure. L'image n'est pas visible. On peut l'afficher avec la fonction [pdf_execute_image\(\)](#), et aussi souvent que désiré. Cela permet d'utiliser plusieurs fois la même image, sans augmenter la taille du fichier. [pdf_put_image\(\)](#) et [pdf_execute_image\(\)](#) est fortement recommandé pour les images de grande taille (plusieurs ko) si elles sont affichées plus d'une fois dans le document. Note : *Cette fonction n'a plus de sens avec la version 2.01 de pdflib. Elle ne fera que retourner une alerte.*

Voir aussi [pdf_put_image\(\)](#), [pdf_place_image\(\)](#), [pdf_execute_image\(\)](#).

10.54.74 [pdf_execute_image](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_execute_image](#)(int *pdf document*, int *image*, double *x-coor*, double *y-coor*, double *scale*)

[PHP 3>= 3.0.7, 4.0b2 – 4.0b4 only]

[pdf_execute_image\(\)](#) affiche une image qui a été enregistrée dans le document PDF, avec la fonction [pdf_put_image\(\)](#). L'image est implantée dans la page courante, et aux coordonnées données.

L'image peut être modifiée d'échelle en même temps qu'elle est affichée. Une échelle de 1.0 affichera l'image à sa taille d'origine. Note : *Cette fonction n'a plus de sens avec la version 2.01 de pdflib. Elle ne fera que retourner une alerte.*

Affichage multiple d'une image.

```
<?php
$im = ImageCreate(100, 100);
$coll = ImageColorAllocate($im, 80, 45, 190);
ImageFill($im, 10, 10, $coll);
$pim = PDF_open_memory_image($pdf, $im);
pdf_put_image($pdf, $pim);
pdf_execute_image($pdf, $pim, 100, 100, 1);
pdf_execute_image($pdf, $pim, 200, 200, 2);
pdf_close_image($pdf, $pim);
?>
```

10.54.75 pdf_add_annotation

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_add_annotation](#)(int *pdf document*, double *llx*, double *lly*, double *urx*, double *ury*, string *title*, string *content*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b2]

[pdf_add_annotation\(\)](#) ajoute une note, dont le coin inférieur droit est (*llx*, *lly*) et le coin supérieur droit est (*urx*, *ury*).

10.54.76 pdf_set_border_style

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_border_style](#)(int *pdf document*, string *style*, double *width*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b2]

[pdf_set_border_style\(\)](#) modifie le style des bords de liens et d'annotation. *style* peut valoir 'solid' (trait plain) ou 'dashed' (pointillé).

Voir aussi [pdf_set_border_color\(\)](#), [pdf_set_border_dash\(\)](#).

10.54.77 pdf_set_border_color

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_border_color](#)(int *pdf document*, double *red*, double *green*, double *blue*)

[PHP 3>= 3.0.12, PHP 4 >= 4.0b2]

[pdf_set_border_color\(\)](#) modifie la couleur des bords de liens et d'annotation. Les trois composants *red*, *green*, *blue* représentent une couleur RGB (rouge, vert, bleu) et leur valeur doit être comprise entre 0.0 et 1.0.

Voir aussi [pdf_set_border_style\(\)](#), [pdf_set_border_dash\(\)](#).

10.54.78 pdf_set_border_dash

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pdf_set_border_dash](#)(int *pdf document*, double *black*, double *white*)

[PHP 4 >= 4.0.1]

[pdf_set_border_dash\(\)](#) modifie la longueur des pointillés (si le style de bord d'une annotation est en pointillé). *black* représente la taille des traits noirs, et *white* celle des espaces blancs.

Voir aussi [pdf_set_border_style\(\)](#), [pdf_set_border_color\(\)](#).

10.55 Verisign Payflow Pro Paiement

[\[Notes en ligne\]](#)

Cette extension vous permet d'effectuer des transactions avec des cartes de crédits en utilisant les services Verisign Payment Services, anciennement connu sous le nom de Signio (<http://www.verisign.com/payment/>).

Ces fonctions sont utilisables dès que PHP a été compilé avec l'option `--with-pfpro[=DIR]`. Vous devez aussi utiliser le SDK approprié sur votre plate-forme : il est disponible [l'interface du manager](#), une fois que vous vous êtes inscrit.

Une fois que vous avez téléchargé le SDK vous devez copier les fichiers depuis le dossier ``lib'` de la distribution. Copier le fichier d'entête ``pfpro.h'` dans ``usr/local/include'` et la librairie ``libpfpro.so'` dans ``usr/local/lib'`.

Lorsque vous utilisez ces fonctions, vous pouvez omettre d'appeler les fonctions [pfpro_init\(\)](#) et [pfpro_cleanup\(\)](#) : l'extension se chargera de le faire automatiquement. Cependant, elles sont toujours disponibles au cas où vous auriez un grand nombre de transaction à traiter, ou que vous souhaiteriez un contrôle plus fin de la librairie. Vous pouvez effectuer autant de transaction que vous le souhaitez avec [pfpro_process\(\)](#) lors d'une connexion.

Ces fonctions ont été ajoutée en PHP 4.0.2.

Note : *Ces fonctions ne font que fournir un accès aux services Verisign Payment Services. Assurez vous bien de lire le "Payflow Pro Developers Guide" pour plus de détails sur les paramètres.*

[10.55.1 pfpro_init](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`void pfpro_init`

[PHP 4 >= 4.0.2]

[pfpro_init\(\)](#) initialise la librairie Payflow Pro library. Vous pouvez omettre cet appel : dans ce cas, elle sera appelée automatiquement [pfpro_init\(\)](#) avant la première transaction.

Voir aussi [pfpro_cleanup\(\)](#).

[10.55.2 pfpro_cleanup](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`void pfpro_cleanup`

[PHP 4 >= 4.0.2]

[pfpro_cleanup\(\)](#) sert à terminer proprement une session avec la librairie Payflow Pro library. Elle doit être appelé après toutes les transactions, et avant la fin du script. Cependant, vous pouvez omettre cet appel : dans ce cas, cette fonction sera automatiquement appelée à la fin du script.

Voir aussi [pfpro_init\(\)](#).

[10.55.3 pfpro_process](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

`array pfpro_process (array parameters, string address, int port, int timeout, string proxy address, int proxy port, string proxy login, string proxy password)`

[PHP 4 >= 4.0.2]

Retourne un tableau associatif, contenant la réponse.

[pfpro_process\(\)](#) effectue une transaction avec Payflow Pro. Le premier paramètre est un tableau associatif contenant des paires clés/valeurs, qui seront encodées, puis passées au serveur.

Le second paramètre *address* indique quel hôte contacter. Il est optionnel. Par défaut, il vaut "test.signio.com" : vous devrez probablement le remplacer par "connect.signio.com" pour effectuer de vraies transactions.

Le troisième paramètre **port** spécifie le port de connexion. Par défaut, c'est 443, le port SSL standard.

Le quatrième paramètre **timeout** indique le temps de timeout à utiliser. Par défaut, c'est 30 secondes. Notez que ce timeout ne prend effet que lorsqu'une connexion a été établie avec un serveur : votre script peut potentiellement attendre indéfiniment un événement DNS ou un problème de réseau.

Le cinquième paramètre **proxy address** indique le nom du proxy SSL. Le sixième paramètre **proxy port** indique le port à utiliser sur ce proxy.

Les septième et huitième paramètres, **proxy login** et **proxy password** indiquent le nom de compte et le mot de passe à utiliser sur le proxy.

Cette fonction retourne un tableau associatif.

Note : Lisez attentivement le "Payflow Pro Developers Guide" pour connaître les détails des autres paramètres.

Exemple Payflow Pro

```
<?php
pfpro_init();
$transaction = array(USER      => 'monlogin',
PWD      => 'mmotdepasse',
TRXTYPE => 'S',
TENDER  => 'C',
AMT     => 1.50,
ACCT    => '4111111111111111',
EXPDATE => '0904'
);
$response = pfpro_process($transaction);
if (!$response) {
die("Impossible d'établir un lien avec Verisign.\n");
}
echo "La réponse de Verisign était ".$response[RESULT];
echo ", c'est à dire : ".$response[RESPMSG]."\n";
echo "\nLa requête de transaction: ";
print_r($transaction);
echo "\nLa réponse: ";
print_r($response);
pfpro_cleanup();
?>
```

10.55.4 pfpro_process_raw

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pfpro_process_raw](#)(string **parameters**, string **address**, int **port**, int **timeout**, string **proxy address**, int **proxy port**, string **proxy login**, string **proxy password**)
[PHP 4 >= 4.0.2]

Retourne une chaîne avec une réponse.

[pfpro_process_raw\(\)](#) envoie une transaction brute au serveur Payflow Pro. Il est vivement recommandé d'utiliser [pfpro_process\(\)](#) à la place, car les règles de codage sont non standard.

Le premier argument est une chaîne contenant la transaction brute. Tous les autres paramètres sont les mêmes que ceux de [pfpro_process\(\)](#). La valeur de retour est une chaîne contenant la réponse brute.

Note : Lisez attentivement le "Payflow Pro Developers Guide" pour connaître tous les détails des paramètres et leur règle d'encodage. Il est recommandé d'utiliser plutôt [pfpro_process\(\)](#).

Exemple pfpro_process_raw()

```
<?php  
pfpro_init();  
$response = pfpro_process("USER=mylogin&PWD[5]=m&ndy&TRXTYPE=S&TENDER=C&AMT=1.50&ACCT=41111111111111111111");  
if (!$response) {  
die("Impossible de contacter Verisign.\n");  
}  
echo "La réponse brute de Verisign est ".$response;  
pfpro_cleanup();  
?>
```

10.55.5 pfpro version

[\[Notes en ligne\]](#) [\[Exemples\]](#)

```
string pfpro version
```

[PHP 4 >= 4.0.2]

`pfpro version()` lit la version de la librairie Payflow Pro. Au moment de rédaction de la doc, c'était L211.

10.56 PostgreSQL

[\[Notes en ligne\]](#)

Postgres, initialement développé au département de Science informatique, à UC Berkeley, mis en place la majorité des concepts des bases relationnelles, actuellement disponibles sur le marché. PostgreSQL accepte le langage SQL92/SQL3, assure l'intégrité transactionnelle, et l'extension de type. PostgreSQL est une évolution du code originale de Berkeley : il est Open Source et dans le domaine public.

PostgreSQL est disponible sans frais. La version actuelle est disponible à (en anglais) :

www.PostgreSQL.org.

Depuis la version 6.3 (03/02/1998) PostgreSQL utilise les sockets UNIX, et une table est dédiée à ces nouvelles capacités. La socket est située dans le dossier ``/tmp/.s.PGSQL.5432'`. Cette option peut être activée avec `-i` passé au `postmaster` et cela s'interprète: "écoute sur les sockets TCP/IP et sur les sockets Unix".

Postmaster	PHP	Statut
postmaster &	pg_connect("dbname=MonDbName");	OK
postmaster -i &	pg_connect("dbname=MonDbName");	OK
postmaster &	pg_connect("host=localhost dbname=MonDbName");	Unable to connect to PostgreSQL server: connectDB() failed: Impossible de se connecter au serveur PostgreSQL: connectDB() a échoué. Est ce que le postmaster fonctionne, et accepte les TCP/IP (option -i) sur le port '5432'? @tab
postmaster -i &	pg_connect("host=localhost	OK

	dbname=MonDbName");	
--	---------------------	--

Il est possible de se connecter avec la commande suivante : `$conn = pg_Connect("host=monHote port=monPort tty=monTTY options=mesOptions user=monUser password=monPassword dbname=maDB");`

L'ancienne syntaxe : `$conn = pg_connect("host", "port", "options", "tty", "dbname")` est obsolète.

Pour utiliser l'interface des grands objets (large object (lo) interface), il est nécessaire de les placer dans un bloc de transaction. Un bloc de transaction commence avec `begin` et si la transaction se termine avec un `commit` et `end`. Si la transaction échoue, elle doit être conclue par un `abort` et `rollback`.

Utilisation des objets de grande taille (Large Objects)

```
<?php
$database = pg_connect("", "", "", "", "jacarta");
pg_exec($database, "begin");
    $oid = pg_locreate($database);
    echo "$oid\n";
    $handle = pg_loopen($database, $oid, "w");
    echo "$handle\n";
    pg_lowrite($handle, "gaga");
    pg_loclose($handle);
    pg_exec($database, "commit")
    pg_exec($database, "end")
?>
```

10.56.1 pg_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_close](#)(int *connection*)

[PHP 3, PHP 4]

Retourne FALSE si l'index de connexion n'est pas valable, et TRUE sinon. Ferme la connexion au serveur PostgreSQL associé à *connection*.

Note : *Il n'est généralement pas nécessaire de fermer une connexion non persistante, car elles sont automatiquement fermées à la fin d'un script.*

[pg_close\(\)](#) ne ferme pas les connexions persistantes ouvertes avec [pg_pconnect\(\)](#).

10.56.2 pg_cmdtuples

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_cmdtuples](#)(int *result_id*)

[PHP 3, PHP 4]

[pg_cmdtuples\(\)](#) retourne le nombre de tuples (instances) affectés par les requêtes INSERT, UPDATE, et DELETE. Si aucun tuple n'a été affecté, la fonction retournera 0.

pg_cmdtuples

```
<?php
```

```
$result = pg_exec($conn, "INSERT INTO verlag VALUES ('Auteur')");
$cmdtuples = pg_cmdtuples($result);
echo $cmdtuples . " <- tuples modifiés.";
?>
```

10.56.3 pg_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_connect](#)(string *conn_string*)

[PHP 3, PHP 4]

conn_string retourne un index de connexion en cas de succès, et FALSE sinon. Ouvre une connexion à un serveur PostgreSQL. Les arguments doivent être placé entre guillemets.

Using pg_connect arguments

```
<?php
$dbconn = pg_connect("dbname=marie");
//connexion à une base de données nommée "marie"
$dbconn2 = pg_connect("host=localhost port=5432 dbname=marie");
//connexion à une base de données nommée "marie" sur l'hôte "localhost" sur le port "5432"
$dbconn3 = pg_connect("user=agneau password=baaaa dbname=marie ");
//connexion à une base de données nommée "marie" avec l'utilisateur "agneau" et le mot de passe "
?>
```

Les arguments disponibles comptent notamment *dbname*, *port*, *host*, *tty*, *options*, *user*, et *password*

[pg_connect\(\)](#) retourne un index de connexion qui sera nécessaire aux autres fonctions PostgreSQL. Vous pouvez ouvrir plusieurs connexions simultanées.

L'ancienne syntaxe `$conn = pg_connect("host", "port", "options", "tty", "dbname")` est obsolète.

Voir aussi [pg_pconnect\(\)](#).

10.56.4 pg_dbname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_dbname](#)(int *connection*)

[PHP 3, PHP 4]

Retourne le nom de la base de donnée PostgreSQL associée à l'index de connexion *connection*, ou FALSE si *connection* n'est pas valide.

10.56.5 pg_end_copy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_end_copy](#)(resource *connection*)

[PHP 4 >= 4.0.3]

[pg_end_copy\(\)](#) synchronise le client PostgreSQL (ici PHP) avec le serveur, après une opération de copie. Il faut utiliser cette fonction, sous peine de recevoir une erreur "out of sync" (désynchronisé). Retourne TRUE en cas de succès, et FALSE sinon.

Pour plus de détails et un exemple voyez : [pg_put_line\(\)](#).

10.56.6 pg_errormessage

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_errormessage](#) (int *connection*)
[PHP 3, PHP 4]

Retourne une chaîne contenant le dernier message d'erreur, ou FALSE en cas d'échec. Il sera impossible d'obtenir des détails sur l'erreur générée, en utilisant la fonction [pg_errormessage\(\)](#) si une erreur est survenue dans la dernière action pour laquelle une connexion valide existe. Cette fonction retournera une chaîne contenant le message d'erreur généré par le serveur final.

10.56.7 pg_exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_exec](#) (int *connection*, string *query*)
[PHP 3, PHP 4]

Retourne un index de résultat, si la requête a été correctement exécutée, et FALSE en cas d'échec, ou si la connexion *connection* n'était pas un index de connexion valide. En cas d'erreur, le message d'erreur peut être obtenu grâce à la fonction [pg_errormessage\(\)](#), si l'index de connexion était valide. Envoie une requête à un serveur PostgreSQL identifié grâce à l'index de connexion. La réponse retournée par cette fonction est un index de résultat qui devra être utilisé pour accéder aux lignes de résultat, grâce à d'autres fonctions PostgreSQL. Note : *PHP/FI* retournait 1 lorsque la requête n'attendait pas de données en réponse (insertion, modifications, par exemple), et retournait un nombre plus grand que 1, même sur un select qui donnait un ensemble vide. Ce n'est plus le cas.

10.56.8 pg_fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [pg_fetch_array](#) (int *result*, int *row*, int *result_type*)
[PHP 3>= 3.0.1, PHP 4]

Retourne un tableau qui contient à la ligne demandée, dans le résultat identifiée par *result*, et FALSE , si il ne reste plus de lignes.

[pg_fetch_array\(\)](#) est une version évoluée de [pg_fetch_row\(\)](#). En plus de proposer un tableau à indice numérique, elle peut aussi enregistrer les données dans un tableau associatif, en utilisant les noms des champs comme clés.

L'argument optionnel *result_type* de [pg_fetch_array\(\)](#) est une constante, qui peut prendre les valeurs suivantes : PGSQL_ASSOC, PGSQL_NUM, et PGSQL_BOTH. Note : *result_type* a été ajoutée en PHP 4.0.

Il est important de noter que [pg_fetch_array\(\)](#) n'est pas significativement plus lent que [pg_fetch_row\(\)](#), tandis qu'elle fournit un confort d'utilisation notable.

Pour plus de détails, reportez vous à [pg_fetch_row\(\)](#).

PostgreSQL fetch array

```
<?php
$conn = pg_pconnect("dbname=publisher");
if (!$conn) {
    echo "Erreur de connexion.\n";
    exit;
}
$result = pg_exec($conn, "SELECT * FROM authors");
if (!$result) {
    echo "Erreur durant la requete.\n";
    exit;
}
$arr = pg_fetch_array($result, 0);
echo $arr[0] . " <- array\n";
$arr = pg_fetch_array($result, 1);
echo $arr["author"] . " <- array\n";
?>
```

[10.56.9 pg_fetch_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [pg_fetch_object](#) (int *result*, int *row*, int *result_type*)
[PHP 3>= 3.0.1, PHP 4]

Retourne un objet dont les membres sont les champs de la ligne demandée, ou FALSE , si il n'y a plus de lignes.

[pg_fetch_object\(\)](#) est similaire à [pg_fetch_array\(\)](#), avec une différence majeure : c'est un objet qui est retourné, au lieu d'un tableau. Par conséquent, cela signifie que vous ne pouvez accéder aux membres qu'avec leur nom, et non plus leur offset (les nombres ne sont pas autorisés comme nom de membre).

L'argument optionnel *result_type* de *result_type* est une constante qui peut prendre les valeurs suivantes : PGSQL_ASSOC, PGSQL_NUM, et PGSQL_BOTH. Note : *result_type* a été ajouté dans PHP 4.0.

Au niveau vitesse, cette fonction est aussi rapide que [pg_fetch_row\(\)](#) et presque aussi rapide que [pg_fetch_row\(\)](#) (la différence est non significative).

Voir aussi: [pg_fetch_array\(\)](#) et [pg_fetch_row\(\)](#).

Postgres fetch object

```
<?php
$database = "verlag";
$db_conn = pg_connect("host=localhost port=5432 dbname=$database");
if (!$db_conn):
?>
    <H1>Connexion impossible à la base postgres <?php echo $database ></H1> <?php
exit;
endif;
$qu = pg_exec($db_conn, "SELECT * FROM verlag ORDER BY autor");
$row = 0; // postgres réclame un compteur de ligne, d'autres bases ne le font pas.
while ($data = pg_fetch_object($qu, $row)):
    echo $data->autor . " (" ;
    echo $data->jahr . ") : " ;
    echo $data->titel . "<BR>";
    $row++;
endwhile;
?>
<PRE><?php
$fields[] = array("autor", "Author");
$fields[] = array("jahr", "Year");
$fields[] = array("titel", "Title");
```

```

$row= 0; // Postgres réclame un compteur de ligne, d'autres bases ne le font pas.
while ($data = pg_fetch_object($qu, $row)):
echo "-----\n";
reset($fields);
while (list($item) = each($fields)):
echo $item[1].": ".$data->$item[0]."\n";
endwhile;
    $row++;
endwhile;
echo "-----\n";
?>
</PRE>
<?php
pg_freeresult($qu);
pg_close($db_conn);
?>

```

10.56.10 pg_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [pg_fetch_row](#) (int *result*, int *row*)
 [PHP 3>= 3.0.1, PHP 4]

Retourne un tableau qui contient les données de la ligne demandée, ou FALSE , si il ne reste plus de lignes. [pg_fetch_row\(\)](#) lit une ligne dans le résultat associé à l'index *result*. La ligne est retournée sous la forme d'un tableau. La ligne est retournée sous la forme d'un tableau, qui commence à l'index 0. Les appels ultérieurs à [pg_fetch_row\(\)](#) retourneront la ligne d'après, ou bien FALSE, lorsqu'il n'y aura plus de lignes.

Voir aussi: [pg_fetch_array\(\)](#), [pg_fetch_object\(\)](#) et [pg_result\(\)](#).

Postgres retourne une ligne

```

<?php
$conn = pg_pconnect("dbname=publisher");
if (!$conn) {
echo "Une erreur est survenue.\n";
exit;
}
$result = pg_exec($conn, "SELECT * FROM authors");
if (!$result) {
echo "Une erreur est survenue.\n";
exit;
}
$num = pg_numrows($result);
for ($i=0; $i<$num; $i++) {
    $r = pg_fetch_row($result, $i);
    for ($j=0; $j<count($r); $j++) {
echo "$r[$j]&nbsp;";
    }
echo "<BR>";
}
?>

```

10.56.11 pg_fieldisnull

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldisnull](#) (int *result_id*, int *row*, mixed *field*)

[PHP 3, PHP 4]

Teste si un champs est à NULL. Retourne 0 si le champs n'est pas NULL. Retourne 1 si le champs est à NULL. Le champs peut être identifié avec son nom ou son index numérique (commencant à 0).

10.56.12 pg_fieldname

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_fieldname](#) (int *result_id*, int *field_number*)

[PHP 3, PHP 4]

[pg_fieldname\(\)](#) va retourner le nom du champs qui occupe la colonne numéro *field_number* dans le résultat *result_id*. La numérotation des champs commence à 0.

10.56.13 pg_fieldnum

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldnum](#) (int *result_id*, string *field_name*)

[PHP 3, PHP 4]

[pg_fieldnum\(\)](#) retourne le numéro de la colonne, dont le nom est *field_name*, dans le résultat *result_id*. La numérotation des champs commence à 0. Cette fonction retournera -1 en cas d'erreur.

10.56.14 pg_fieldprtlen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldprtlen](#) (int *result_id*, int *row_number*, string *field_name*)

[PHP 3, PHP 4]

[pg_fieldprtlen\(\)](#) retourne la taille imprimée (nombre de caractères) d'une valeur donnée dans un résultat PostgreSQL. La numérotation des lignes commence à 0. Cette fonction retourne -1 en cas d'erreur.

10.56.15 pg_fieldsize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_fieldsize](#) (int *result_id*, int *field_number*)

[PHP 3, PHP 4]

[pg_fieldsize\(\)](#) retourne la taille interne de stockage d'un champs donné, en octets. Retourne -1 si la taille est variable. Retourne FALSE en cas d'erreur. La numérotation des colonnes commence à 0.

10.56.16 pg_fieldtype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_fieldtype](#) (int *result_id*, int *field_number*)

[PHP 3, PHP 4]

[pg_fieldtype\(\)](#) retourne une chaîne contenant le type du champs donné par son index *field_number* . La numérotation des champs commence à 0.

10.56.17 pg_freeresult

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_freeresult](#) (int *result_id*)

[PHP 3, PHP 4]

[pg_freeresult\(\)](#) n'est vraiment utile que si vous risquez d'utiliser trop de mémoire durant votre script. La mémoire occupée par les résultats est automatiquement libérée à la fin du script. Mais, si vous êtes sûr de ne pas avoir besoin du résultat ultérieurement, vous pouvez appeler [pg_freeresult\(\)](#) avec l'index de résultat comme argument, et la mémoire sera libérée.

10.56.18 pg_getlastoid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_getlastoid](#) (int *result_id*)

[PHP 3, PHP 4]

[pg_getlastoid\(\)](#) sert à lire l' Oid assigné à un tuple inséré, si l'index de résultat a été obtenu avec la fonction [pg_exec\(\)](#), dont la requête était exclusivement SQL INSERT. Cette fonction retourne un entier positif si un Oid valide a été trouvé. Elle retournera -1 si une erreur est survenue, ou si la dernière commande n'était pas un INSERT.

10.56.19 pg_host

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_host](#) (int *connection_id*)

[PHP 3, PHP 4]

[pg_host\(\)](#) retourne le nom d'hôte associé à l'index de connexion PostgreSQL.

10.56.20 pg_loclose

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [pg_loclose](#) (int *fd*)

[PHP 3, PHP 4]

[pg_loclose\(\)](#) ferme un objet de type Inversion Large Object. *fd* est un descripteur de fichier, obtenu avec [pg_loopen\(\)](#).

10.56.21 pg_locreate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_locreate](#)(int *conn*)

[PHP 3, PHP 4]

[pg_locreate\(\)](#) crée un objet de type Inversion Large Object et retourne son Oid. *conn* doit être une connexion valide avec une base de données PostgreSQL. Les modes d'accès PostgreSQL INV_READ, INV_WRITE, et INV_ARCHIVE ne sont pas supportés : l'objet peut toujours être créé, avec des droits d'accès en lecture et écriture. Le mode INV_ARCHIVE a été supprimé des bases PostgreSQL (version 6.3 et ultérieur).

10.56.22 pg_loexport

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_loexport](#)(int *oid* , int *file* , int *connection_id*)

[PHP 4 >= 4.0.1]

oid est un identifiant d'objet de grande taille qui sera exporté dans le fichier *filename*, qui spécifie son chemin. Retourne FALSE si une erreur survient, et TRUE en cas de succès. N'oubliez pas que la manipulation d'un objet de grande taille dans PostgreSQL doit intervenir dans une transaction.

10.56.23 pg_loimport

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_loimport](#)(int *file* , int *connection_id*)

[PHP 4 >= 4.0.1]

filename est le chemin jusqu'à un fichier qui servira de source pour créer un objet de grande taille. La fonction retourne FALSE en cas d'erreur, et sinon un identifiant d'objet, créé directement à la bonne taille. N'oubliez pas que la manipulation d'un objet de grande taille dans PostgreSQL doit intervenir dans une transaction.

10.56.24 pg_loopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_loopen](#)(int *conn*, int *objoid*, string *mode*)

[PHP 3, PHP 4]

[pg_loopen\(\)](#) ouvre un objet de type Inversion Large Object et retourne un descripteur de fichier pour cet objet. Le descripteur de fichier contient les informations de connexion. Ne refermez pas la connexion avant d'avoir fermé l'objet. *objoid* est un Oid valide de Large Object, et *mode* peut prendre es valeurs suivantes : "r", "w", ou "rw".

10.56.25 pg_loread

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_loread](#)(int *fd*, int *len*)

[PHP 3, PHP 4]

[`pg_loread\(\)`](#) lit au plus *len* octets d'un objet de grande taille, et retourne les données sous la forme d'une chaîne. *fd* est un identifiant valide d'objet de grande taille, et *len* indique la taille maximale de mémoire alloué à l'objet de grande taille.

[10.56.26 pg_loreadall](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`pg_loreadall`](#)(int *fd*)

[PHP 3, PHP 4]

[`pg_loreadall\(\)`](#) lit un objet de grande taille en totalité et le passe directement au client, après les entêtes adéquats. Cette fonction est prévue pour transmettre des sons ou des images.

[10.56.27 pg_lounlink](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`pg_lounlink`](#)(int *conn*, int *lobjid*)

[PHP 3, PHP 4]

[`pg_lounlink\(\)`](#) efface l'objet de grande taille dont l'identifiant est *lobjid*.

[10.56.28 pg_lowrite](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`pg_lowrite`](#)(int *fd*, string *buf*)

[PHP 3, PHP 4]

[`pg_lowrite\(\)`](#) écrit dans l'objet de grande taille autant de données possible, issues de la variable *buf* et retourne le nombre d'octets réellement écrits, ou FALSE en cas d'erreur. *fd* est un descripteur d'objet de grande taille, obtenu avec [`pg_loopen\(\)`](#).

[10.56.29 pg_numfields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`pg_numfields`](#)(int *result_id*)

[PHP 3, PHP 4]

[`pg_numfields\(\)`](#) retourne le nombre de champs ou (colonnes) d'un résultat PostgreSQL. L'argument doit être un identifiant de résultat valide retourné par [`pg_exec\(\)`](#). Cette fonction retournera -1 en cas d'erreur.

[10.56.30 pg_numrows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`pg_numrows`](#)(int *result_id*)

[PHP 3, PHP 4]

[pg_numrows\(\)](#) retourne le nombre de lignes d'un résultat PostgreSQL. L'argument doit être un identifiant de résultat valide retourné par [pg_exec\(\)](#). Cette fonction retournera -1 en cas d'erreur.

[10.56.31 pg_options](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_options](#) (int *connection_id*)
[PHP 3, PHP 4]

[pg_options\(\)](#) retourne une chaîne contenant les options de la connexion PostgreSQL.

[10.56.32 pg_pconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_pconnect](#) (string *conn_string*)
[PHP 3, PHP 4]

Retourne un index de connexion en cas de succès, ou FALSE en cas d'erreur. Ouvre une connexion permanente à une base PostgreSQL. Les arguments doivent être insérés dans une chaîne à guillemets. Ils incluent : *dbname port, host, tty, options, user*, et *password*

[pg_pconnect\(\)](#) retourne un identifiant de connexion qui sera utilisées par les autres fonctions PostgreSQL. Vous pouvez ouvrir plusieurs connexions en même temps.

L'ancienne syntaxe `$conn = pg_pconnect("host", "port", "options", "tty", "dbname")` est obsolète.

[10.56.33 pg_port](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_port](#) (int *connection_id*)
[PHP 3, PHP 4]

[pg_port\(\)](#) retourne le numéro de port de la connexion identifiée *connection_id*.

[10.56.34 pg_put_line](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_put_line](#) (resource *connection_id* , string *data*)
[PHP 4 >= 4.0.3]

[pg_put_line\(\)](#) envoie une chaîne (terminée par NULL) au serveur PostgreSQL. Ceci est pratique pour effectuer des insertions très rapides dans une table, initiée par une opération de copie PostgreSQL copy-operation. Le caractère final NULL est automatiquement ajouté. Retourne TRUE en cas de succès, et FALSE.

Note : *Notez que l'application doit explicitement ajouter les deux caractères "\." à la fin de la chaîne pour indiquer au serveur qu'elle a fini d'envoyer des données.*

Voir aussi [pg_end_copy\(\)](#).

Insertion à grande vitesse dans une table

```
<?php
    $conn = pg_pconnect("dbname=foo");
    pg_exec($conn, "create table bar (a int4, b char(16), d float8)");
    pg_exec($conn, "copy bar from stdin");
    pg_put_line($conn, "3\tBonjour le monde\t4.5\n");
    pg_put_line($conn, "4\tAu revoir le monde\t7.11\n");
    pg_put_line($conn, "\\.\n");
    pg_end_copy($conn);
?>
```

10.56.35 pg_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [pg_result](#)(int *result_id*, int *row_number*, mixed *fieldname*)
[PHP 3, PHP 4]

[pg_result\(\)](#) retourne les valeurs d'un identifiant de résultat, produit par [pg_exec\(\)](#). Les arguments *row_number* et *fieldname* précisent la cellule qui sera retournée. La numérotation des lignes commence à 0. Au lieu d'utiliser le nom du champs, vous pouvez utiliser son index, sous la forme d'un nombre sans guillemets. La numérotation des champs commence à 0.

PostgreSQL dispose de nombreux types, et seuls, les types basiques sont supportés ici. Toutes les formes d'entier, booléen et Oid sont retournés sous la forme d'entiers. Toutes les formes de nombre à virgule flottante et types réels sont retournés sous la forme d'une valeur de type double. Tous les autres types, y compris les tableaux, sont retournés sous la forme de chaînes formatées, au format par défaut de PostgreSQL.

10.56.36 pg_set_client_encoding

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pg_set_client_encoding](#)(int *connection* , string *encoding*)
[PHP 4 >= 4.0.3]

[pg_set_client_encoding\(\)](#) fixe l'encodage du client. Elle retourne 0 en cas de succès, et -1 sinon. *encoding* est l'encodage du client, et peut être SQL_ASCII, EUC_JP, EUC_CN, EUC_KR, EUC_TW, UNICODE, MULE_INTERNAL, LATINX (X=1...9), KOI8, WIN, ALT, SJIS, BIG5, WIN1250.

Note : Cette fonction requiert PHP-4.0.2 ou plus récent et PostgreSQL-7.0 ou plus récent.

Jadis, [pg_set_client_encoding\(\)](#) s'appelait `pg_setclientencoding()`.

Voir aussi [pg_client_encoding\(\)](#).

10.56.37 pg_client_encoding

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_client_encoding](#)(int *connection*)
[PHP 4 >= 4.0.3]

[pg_client_encoding\(\)](#) retourne l'encodage du client. Elle retourne une des valeurs suivantes : SQL_ASCII, EUC_JP, EUC_CN, EUC_KR, EUC_TW, UNICODE, MULE_INTERNAL, LATINX (X=1...9), KOI8, WIN, ALT, SJIS, BIG5, WIN1250.

Note : Cette fonction requiert PHP-4.0.2 ou plus récent et PostgreSQL-7.0 ou plus récent.

Jadis, [pg_client_encoding\(\)](#) s'appelait `pg_clientencoding()`.

Voir aussi [pg_set_client_encoding\(\)](#).

[10.56.38 pg_trace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_trace](#)(string *filename* , string *mode* , int *connection*)

[PHP 4 >= 4.0.1]

Active le suivi des communication entre PHP et le serveur PostgreSQL. Cet historique sera enregistré dans un fichier. Pour comprendre ces lignes, il faut être familier avec le protocole de communication interne à PostgreSQL. Pour ceux qui le ne sont pas, elles peuvent être utiles pour suivre les requêtes et les erreurs : avec la commande `grep '^To backend' trace.log`, vous pourrez voir les requêtes réellement envoyées au serveur PostgreSQL.

filename et *mode* sont les mêmes arguments que pour la fonction [fopen\(\)](#) (*mode* par défaut à 'w'), *connection* indique la connexion à suivre. Par défaut, c'est la dernière ouverte.

Retourne TRUE si *filename* a pu être ouvert en écriture, et FALSE sinon.

Voir aussi [fopen\(\)](#) et [pg_untrace\(\)](#).

[10.56.39 pg_tty](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [pg_tty](#)(int *connection_id*)

[PHP 3, PHP 4]

[pg_tty\(\)](#) retourne le nom de tty de la connexion associée à *connection_id*.

[10.56.40 pg_untrace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [pg_untrace](#)(int *connection*)

[PHP 4 >= 4.0.1]

Termine le suivi d'une connexion PostgreSQL, initiée avec [pg_trace\(\)](#). *connection* indique la connexion à suivre. Par défaut, c'est la dernière ouverte.

Retourne toujours TRUE.

Voir aussi [pg_trace\(\)](#).

[10.57 POSIX](#)

[\[Notes en ligne\]](#)

Ce module contient une interface avec les documents au standard IEEE 1003.1 (POSIX.1), qui ne sont pas accessibles autrement. Par exemple, POSIX.1 définit les fonctions `open()`, `read()`, `write()` et `close()`, qui ont été traditionnellement les fonctions de PHP 3. Certains fonctionnalités spécifiques ne sont pas encore disponibles, bien que ce module tâche de remédier à cette situation avec ses fonctions.

10.57.1 posix_kill

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_kill](#)(int *pid*, int *sig*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[posix_kill\(\)](#) envoie le signal *sig* au processus *pid*. Retourne FALSE, si il n'a pas pu envoyer le signal, et TRUE sinon.

Reportez vous à la page de manuel de kill(2) de votre système POSIX, qui contient plus de détails sur les identifiants négatifs de processus, les pid spéciaux 0 et -1, et le signal numéro 0.

10.57.2 posix_getpid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getpid](#)(void)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'identifiant du processus courant.

10.57.3 posix_getppid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getppid](#)(void)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'identifiant du processus parent du processus courant.

10.57.4 posix_getuid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getuid](#)(void)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'ID numérique de l'utilisateur du processus courant. Reportez vous à [posix_getpwuid\(\)](#) pour accéder au nom d'utilisateur.

10.57.5 posix_geteuid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_geteuid](#)(void)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'UID effectif de l'utilisateur du processus courant. Reportez vous à [posix_getpwuid\(\)](#) pour obtenir le nom d'utilisateur.

[10.57.6 posix_getgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getgid](#)(void)
[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'UID du groupe du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour accéder au nom du groupe.

[10.57.7 posix_getegid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getegid](#)(void)
[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'ID effectif du groupe du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour transformer cette information en nom de groupe.

[10.57.8 posix_setuid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_setuid](#)(int *uid*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Fixe l'UID effective de l'utilisateur du processus courant. Vous devez avoir les privilèges nécessaires (traditionnellement ceux du root) sur votre système pour faire ceci.

Retourne TRUE en cas de succès, FALSE sinon. Voir aussi [posix_setgid\(\)](#).

[10.57.9 posix_setgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_setgid](#)(int *gid*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Fixe le GID effective du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour transformer cette information en nom de groupe. L'ordre approprié est d'abord [posix_setgid\(\)](#), puis [posix_setuid\(\)](#).

Retourne TRUE en cas de succès, FALSE sinon.

[10.57.10 posix_getgroups](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getgroups](#)(void)
[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne un tableau contenant les identifiants du groupe du processus courant. Reportez vous à [posix_getgrgid\(\)](#) pour pouvoir utiliser ces id.

[10.57.11 posix_getlogin](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_getlogin](#)(void)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Retourne le nom de login de l'utilisateur qui possède le processus courant. Reportez vous à [posix_getpwnam\(\)](#) pour obtenir plus d'informations sur cet utilisateur.

[10.57.12 posix_getpgrp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getpgrp](#)(void)
[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'identifiant du groupe de processus du processus courant. Reportez vous à POSIX.1 et à [getpgrp\(2\)](#) dans le manuel de votre système POSIX pour plus d'informations sur les groupes de processus.

[10.57.13 posix_setsid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_setsid](#)(void)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Fait du processus courant un chef de session. Reportez vous à POSIX.1 et [setsid\(2\)](#) dans le manuel de votre système POSIX pour plus d'informations sur le contrôle de tâche. Retourne un identifiant de session.

[10.57.14 posix_setpgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_setpgid](#)(int *pid*, int *pgid*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

[posix_setpgid\(\)](#) ajoute le processus *pid* au groupe d'id *pgid*. Reportez vous à POSIX.1 et [setsid\(2\)](#) dans le manuel de votre système POSIX pour plus d'informations sur le contrôle de tâche. Retourne TRUE en cas de succès, et FALSE sinon.

[10.57.15 posix_getpgid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getpgid](#)(int *pid*)
[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne l'id du groupe de processus pour le processus *pid*.

Ceci n'est pas une fonction POSIX, mais elle est répandue sur les systèmes BSD et System V. Si votre système ne supporte pas cette fonction, la fonction PHP retournera toujours FALSE.

[10.57.16 posix_getsid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [posix_getsid](#)(int *pid*)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne le sid du processus *pid*. Si *pid* est à 0, le sid retourné sera celui du processus courant.

Ceci n'est pas une fonction POSIX, mais elle est répandue sur les systèmes BSD et System V. Si votre système ne supporte pas cette fonction, la fonction PHP retournera toujours FALSE.

[10.57.17 posix_uname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_uname](#)(void)

[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Retourne un tableau associatif avec des informations sur le système. Les indices du tableau sont :

- sysname – nom du système d'exploitation (e.g. Linux)
- nodename – nom du système (e.g. valiant)
- release – édition du système d'exploitation (e.g. 2.2.10)
- version – version du système d'exploitation (e.g. #4 Tue Jul 20 17:01:36 MEST 1999)
- machine – architecture système (e.g. i586)

Posix impose que vous n'ayez pas d'a priori sur le format des chaînes, c'est à dire que vous ne devez pas vous attendre à avoir forcément 3 chiffres pour la version, par exemple.

[10.57.18 posix_times](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_times](#)(void)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Retourne un tableau avec les informations sur l'utilisation du CPU. Les indices sont :

- ticks – nombre de ticks depuis le dernier démarrage
- utime – temps utilisateur utilisé par le processus courant.
- stime – temps système utilisé par le processus courant.
- cutime – temps utilisateur utilisé par le processus courant et ses enfants.
- cstime – temps système utilisé par le processus courant et ses enfants.

[10.57.19 posix_ctermid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_ctermid](#)(void)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire.

[10.57.20 posix_ttyname](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_ttyname](#) (int *fd*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire.

[10.57.21 posix_isatty](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_isatty](#) (int *fd*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire.

[10.57.22 posix_getcwd](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [posix_getcwd](#) (void)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire très rapidement.

[10.57.23 posix_mkfifo](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [posix_getcwd](#) (string *pathname*, int *mode*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire très rapidement.

[10.57.24 posix_getgrnam](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getgrnam](#) (string *name*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire.

10.57.25 posix_getgrgid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getgrgid](#) (int *gid*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Encore à faire.

10.57.26 posix_getpwnam

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getpwnam](#) (string *username*)
[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Retourne un tableau associatif qui contient des informations à propos d'un utilisateur, identifié par son nom, passé en paramètre *username*.

Les éléments du tableau sont :

Élément	Description
name	Le nom contient le nom de l'utilisateur. Généralement, c'est un nom court, de moins de 16 caractères, mais ce n'est pas son nom réel et complet. Cette valeur devrait correspondre au paramètre <i>username</i> , et donc, il est redondant. @tab
passwd	Contient le mot de passe de l'utilisateur, encrypté. Souvent, dans les systèmes utilisant les mots de passe "fantômes", un astérisque est retourné. @tab
uid	L'UID de l'utilisateur. @tab
gid	L'ID du groupe de l'utilisateur. Utilisez la fonction posix_getgrgid() pour connaître le nom du groupe, et ses membres. @tab
gecos	GECOS est un terme obsolète qui fait référence aux données de finger, sur un système Honeywell. Le champs, cependant, a survécu, et son contenu a été formalisé par POSIX. Le champs contient une liste, séparée par des virgules, qui contient le nom complet de l'utilisateur, son téléphone professionnel, son numéro de téléphone bureau, et son numéro de téléphone personnel. Sur la plus part des systèmes, seul le nom est disponible. @tab
dir	Cet élément contient le chemin absolu jusqu'au dossier racine de l'utilisateur.

	@tab
shell	Cet élément contient le chemin absolu jusqu'au dossier d'exécution du shell de l'utilisateur. @tab

10.57.27 posix_getpwuid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getpwuid](#) (int *uid*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0b4]

Retourne un tableau associatif contenant des informations sur un utilisateur repéré par son UID, passé dans le paramètre *uid*.

Les éléments du tableau sont :

Elément	Description
name	Le nom contient le nom de l'utilisateur. Généralement, c'est un nom court, de moins de 16 caractères, mais ce n'est pas son nom réel et complet. @tab
passwd	Contient le mot de passe de l'utilisateur, encrypté. Souvent, dans les systèmes utilisant les mots de passes "fantômes", un astérisque est retourné. @tab
uid	Cette valeur devrait correspondre au paramètre <i>uid</i> , et donc, il est redondant. @tab
gid	L'ID du groupe de l'utilisateur. Utilisez la fonction posix_getgrgid() pour connaître le nom du groupe, et ses membres. @tab
gecos	GECOS est un terme obsolète qui fait référence aux données de finger, sur un système Honeywell. Le champs, cependant, a survécu, et son contenu a été formalisé par POSIX. Le champs contient une liste, séparée par des virgules, qui contient le nom complet de l'utilisateur, son téléphone professionne, son numéro de bureau, et son numéro de téléphone personnel. Sur la plus part des sytèmes, seul le nom est disponible. @tab
dir	Cet élément contient le chemin absolu jusqu'au dossier racine de l'utilisateur. @tab

shell	Cet élément contient le chemin absolu jusqu'au dossier d'exécution du shell de l'utilisateur. @tab
-------	--

[10.57.28 posix_getrlimit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [posix_getrlimit](#) (void)
[PHP 3>= 3.0.10, PHP 4 >= 4.0b4]

Encore à faire rapidement.

[10.58 Pspell](#)

[\[Notes en ligne\]](#)

La librairie pspell vous permet de vérifier l'orthographe d'un mot, et suggérer des corrections. Vous aurez besoin des librairies aspell et pspell, disponibles à <http://aspell.sourceforge.net/> et <http://pspell.sourceforge.net/> (respectivement). Il faut aussi ajouter l'option `--with-pspell[=dir]` lors de la compilation de PHP.

[10.58.1 pspell_add_to_personal](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_add_to_personal](#) (int *dictionary_link*, string *word*)
[PHP 4 >= 4.0.2]

[pspell_add_to_personal\(\)](#) ajoute un mot au dictionnaire personnel. Si vous utilisez [pspell_new_config\(\)](#) avec [pspell_config_personal\(\)](#) pour ouvrir le dictionnaire, vous pourrez sauver le dictionnaire personnel ultérieurement avec [pspell_save_wordlist\(\)](#). Notez bien que cette fonction ne fonctionnera pas avec les versions antérieures pspell .11.2 et aspell .32.5.

Exemple avec [pspell_add_to_personal\(\)](#)

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
$pspell_link = pspell_new_config ($pspell_config);
pspell_add_to_personal ($pspell_link, "Vlad");
pspell_save_wordlist ($pspell_link);
?>
```

[10.58.2 pspell_add_to_session](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_add_to_session](#)(int *dictionary_link*, string *word*)

[PHP 4 >= 4.0.2]

[pspell_add_to_session\(\)](#) ajoute un mot au dictionnaire personnel associé à la version courante. C'est une fonction similaire à [pspell_add_to_personal\(\)](#).

10.58.3 [pspell_check](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [pspell_check](#)(int *dictionary_link*, string *word*)

[PHP 4 >= 4.0.2]

[pspell_check\(\)](#) vérifie l'orthographe d'un mot et retourne TRUE si l'orthographe est correcte, FALSE sinon.

pspell_check()

```
<?php
$pspell_link = pspell_new ("french");
if (pspell_check ($pspell_link, "testt")) {
    echo "L'orthographe est exacte";
} else {
    echo "Désolé, mauvaise orthographe";
}
?>
```

10.58.4 [pspell_clear_session](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_clear_session](#)(int *dictionary_link*)

[PHP 4 >= 4.0.2]

[pspell_clear_session\(\)](#) remet à zéro la session courante. Le dictionnaire personnel est vidé, et par exemple si vous tentez de l'enregistrer avec [pspell_save_wordlist\(\)](#), rien ne se passera.

Exemple avec [pspell_add_to_personal\(\)](#)

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
$pspell_link = pspell_new_config ($pspell_config);
pspell_add_to_personal ($pspell_link, "Vlad");
pspell_clear_session ($pspell_link);
pspell_save_wordlist ($pspell_link);    //"Vlad" ne sera pas sauvé
?>
```

10.58.5 [pspell_config_create](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_create](#) (string *language*, string *spelling* , string *jargon* , string *encoding*)

[PHP 4 >= 4.0.2]

[pspell_config_create\(\)](#) a une syntaxe similaire à [pspell_new\(\)](#). En fait, utiliser [pspell_config_create\(\)](#) suivi immédiatement par [pspell_new_config\(\)](#) produira exactement le même résultat. Cependant, après avoir créer une nouvelle configuration, vous pouvez aussi utiliser les fonctions `pspell_config_*` avant d'appeler [pspell_new_config\(\)](#) pour tirer profit des fonctionnalités avancées.

Le paramètre de langage est le code de langue en deux lettres, défini dans la norme ISO 639, et deux lettres optionnelles ISO 3166, après un tiret ou un souligné (_).

Le paramètre d'orthographe *spelling* est nécessaire pour les langues qui ont plus d'une orthographe, comme l'anglais. Les valeurs reconnues sont alors 'american' (américain) , 'british' (anglais), et 'canadian' (canadien).

Le paramètre de jargon *jargon* contient des informations supplémentaires pour distinguer deux dictionnaires distincts pour la même langue et le même paramètre d'orthographe *spelling*.

Le paramètre d'encodage indique l'encodage attendu pour la réponse. Les valeurs valides sont : 'utf-8', 'iso8859-*', 'koi8-r', 'viscii', 'cp1252', 'machine unsigned 16', 'machine unsigned 32'. Ce paramètre n'a pas été testé de manière exhaustive, alors soyez prudent.

Le paramètre de mode est le mode de travail du vérificateur d'orthographe. Plusieurs modes sont disponibles :

- PSPELL_FAST – Mode rapide (moins de suggestions, plus de vitesse)
- PSPELL_NORMAL – Mode normal mode (plus de suggestions)
- PSPELL_BAD_SPELLERS – Mode lent (beaucoup plus de suggestions, moins de vitesse)

Pour plus d'informations et d'exemples, vérifiez le manuel pspell sur leur site web

: <http://pspell.sourceforge.net/>.

Exemple avec [pspell_config_create\(\)](#)

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
pspell_config_repl ($pspell_config, "/var/dictionaries/custom.repl");
$pspell_link = pspell_new_personal ($pspell_config);
?>
```

10.58.6 [pspell_config_ignore](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_ignore](#) (int *dictionary_link*, int *n*)

[PHP 4 >= 4.0.2]

[pspell_config_ignore\(\)](#) doit être utilisé avec une configuration avec d'appeler [pspell_new_config\(\)](#). Cette fonction permet au vérificateur d'ignorer les mots trop courts.

Exemple avec `pspell_config_ignore()`

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_ignore($pspell_config, 5);
$pspell_link = pspell_new_config($pspell_config);
pspell_check($pspell_link, "abcd");      // Ce mot ne provoquera pas d'erreur
?>
```

10.58.7 `pspell_config_mode`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_mode](#)(int *dictionary_link*, int *mode*)

[PHP 4 >= 4.0.2]

[pspell_config_mode\(\)](#) doit être appelé avant [pspell_new_config\(\)](#). Cette fonction détermine le nombre de suggestions qui seront retournés par [pspell_suggest\(\)](#).

Le paramètre de mode est le mode de travail du vérificateur d'orthographe. Plusieurs modes sont disponibles :

- PSPELL_FAST – Mode rapide (moins de suggestions, plus de vitesse)
- PSPELL_NORMAL – Mode normal mode (plus de suggestions)
- PSPELL_BAD_SPELLERS – Mode lent (beaucoup plus de suggestions, moins de vitesse)

pspell_config_mode()

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_mode($pspell_config, PSPELL_FAST);
$pspell_link = pspell_new_config($pspell_config);
pspell_check($pspell_link, "thecat");
?>
```

10.58.8 `pspell_config_personal`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_personal](#)(int *dictionary_link*, string *file*)

[PHP 4 >= 4.0.2]

[pspell_config_personal\(\)](#) doit être appelé dans une configuration avant d'appeler [pspell_new_config\(\)](#). Le dictionnaire personnel sera chargé est utilisé en plus du dictionnaire standard, une fois que vous aurez appelé [pspell_new_config\(\)](#). Si le fichier n'existe pas, il sera créé. Ce fichier sera aussi le fichier où [pspell_save_wordlist\(\)](#) sauvera le dictionnaire personnel. Ce fichier devra donc être accessible en écriture par PHP. Notez que cette fonction ne fonctionne pas avec les versions antérieures à pspell .11.2 et aspell .32.5.

Exemple avec `pspell_config_personal()`

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
$pspell_link = pspell_new_config ($pspell_config);
pspell_check ($pspell_link, "thecat");
?>
```

[10.58.9 pspell_config_repl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_repl](#) (int *dictionary_link*, string *file*)

[PHP 4 >= 4.0.2]

[pspell_config_repl\(\)](#) doit être appelé dans une configuration avant d'appeler [pspell_new_config\(\)](#). Les paires de remplacement améliorent la qualité du vérificateur. Lorsqu'un mot est mal orthographié et qu'aucune suggestion valable n'est trouvée dans le dictionnaire, [pspell_store_replacement\(\)](#) sera utilisé pour enregistrer une paire de remplacement et [pspell_save_wordlist\(\)](#) pour sauver le dictionnaire avec les paires de remplacement. Ce fichier devra donc être accessible en écriture par PHP. Notez que cette fonction ne fonctionne pas avec les versions antérieures à pspell .11.2 et aspell .32.5.

Exemple avec pspell_config_repl()

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
pspell_config_repl ($pspell_config, "/var/dictionaries/custom.repl");
$pspell_link = pspell_new_config ($pspell_config);
pspell_check ($pspell_link, "thecat");
?>
```

[10.58.10 pspell_config_runtogether](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_runtogether](#) (int *dictionary_link*, boolean *flag*)

[PHP 4 >= 4.0.2]

[pspell_config_runtogether\(\)](#) doit être appelé dans une configuration avant d'appeler [pspell_new_config\(\)](#). Cette fonction indique si deux mots accolés doivent être traités comme un composé valide, même si il devrait y avoir un espace entre ces deux mots. Modifier cette configuration n'affecte que les résultats retournés par [pspell_check\(\)](#); [pspell_suggest\(\)](#) retournera toujours des suggestions.

Exemple avec pspell_config_runtogether()

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_runtogether ($pspell_config, TRUE);
$pspell_link = pspell_new_config ($pspell_config);
pspell_check ($pspell_link, "thecat");
?>
```

[10.58.11 pspell_config_save_repl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_config_save_repl](#)(int *dictionary_link*, boolean *flag*)

[PHP 4 >= 4.0.2]

[pspell_config_save_repl\(\)](#) doit être appelé dans une configuration avant d'appeler [pspell_new_config\(\)](#). Elle détermine si [pspell_save_wordlist\(\)](#) doit sauver les paires de remplacement avec le dictionnaire. Généralement, il n'y a pas besoin d'utiliser cette fonction car si [pspell_config_repl\(\)](#) est utilisée, les paires de remplacement seront sauvées de toutes façons, et si ce n'est pas le cas, elles ne seront pas sauvées. Ce fichier devra donc être accessible en écriture par PHP. Notez que cette fonction ne fonctionne pas avec les versions antérieures à pspell .11.2 et aspell .32.5.

[10.58.12 pspell_new](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_new](#)(string *language*, string *spelling* , string *jargon* , string *encoding*)

[PHP 4 >= 4.0.2]

[pspell_new\(\)](#) ouvre un nouveau dictionnaire et retourne un identifiant de dictionnaire, pour utiliser avec d'autres fonctions pspell.

Le paramètre de langue *spelling* est constitué des deux lettres du codage de langue ISO 639, et du codage optionnel de pays ISO 3166, séparé par un '_'.

Ce paramètre est nécessaire pour les langues qui ont plus d'une orthographe, comme l'anglais ou le français. Les valeurs reconnues sont ``americain'', ``britannique'', et ``canadien''.

Le paramètre de jargon contient des informations supplémentaires pour distinguer deux listes de mots qui ont le même marquage de langue et d'orthographe.

Le paramètre d'encodage est le type d'encodage des mots. Les valeurs valides sont 'utf-8', 'iso8859-*', 'koi8-r', 'viscii', 'cp1252', 'machine unsigned 16', 'machine unsigned 32'.

Le paramètre de mode est le mode de travail du vérificateur d'orthographe. Plusieurs modes sont disponibles :

- PSPELL_FAST – Mode rapide (moins de suggestions, plus de vitesse)
- PSPELL_NORMAL – Mode normal mode (plus de suggestions)
- PSPELL_BAD_SPELLERS – Mode lent (beaucoup plus de suggestions, moins de vitesse)
- PSPELL_RUN_TOGETHER – Considère que des mots accolés forment un composé autorisé. C'est à dire que "lechat" sera un composé valide. Cette option ne modifie que les résultats retournés par [pspell_check\(\)](#); [pspell_suggest\(\)](#) retournera toujours les mêmes suggestions.

Mode est un champs de bit, construits à partir des constantes listées ci dessus. Cependant, PSPELL_FAST, PSPELL_NORMAL et PSPELL_BAD_SPELLERS sont mutuellement exclusives : vous ne devez en utiliser qu'une seule en même temps.

Pour plus d'informations et d'exemples, reportez vous au site <http://pspell.sourceforge.net/> (en anglais).

pspell_new()

```
<?php
$pspell_link = pspell_new("en", "", "", "", (PSPELL_FAST|PSPELL_RUN_TOGETHER));
?>
```

10.58.13 [pspell_new_config](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_new_config](#) (int *config*)

[PHP 4 >= 4.0.2]

[pspell_new_config\(\)](#) ouvre un nouveau dictionnaire et charge les paramètres spécifiés dans la configuration *config*, créée avec [pspell_config_create\(\)](#) et modifiée avec les fonctions `pspell_config_*`. Cette méthode vous donne le maximum de flexibilité, et dispose de toutes les fonctionnalités fournies par [pspell_new\(\)](#) et [pspell_new_personal\(\)](#).

Le paramètre de configuration est celui qui a été retourné par [pspell_config_create\(\)](#) lors de création de la configuration.

pspell_new_config()

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
pspell_config_repl ($pspell_config, "/var/dictionaries/custom.repl");
$pspell_link = pspell_new_personal (pspell_config);
?>
```

10.58.14 [pspell_new_personal](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_new_personal](#) (string *personal*, string *language*, string *spelling* , string *jargon* , string *encoding* , int *mode*)

[PHP 4 >= 4.0.2]

[pspell_new_personal\(\)](#) charge un nouveau dictionnaire avec un dictionnaire personnel, et retourne un identifiant de dictionnaire utilisé par d'autres fonctions pspells. Le dictionnaire peut être modifié et sauvé avec [pspell_save_wordlist\(\)](#). Cependant, les paires de remplacement ne seront pas sauvées. Pour ce faire, vous devez créer une configuration qui utilise [pspell_config_create\(\)](#), et choisir le fichier de destination du dictionnaire personnel avec [pspell_config_personal\(\)](#), choisir le fichier de paire de remplacement avec [pspell_config_repl\(\)](#), et ouvrir un nouveau dictionnaire avec [pspell_new_config\(\)](#).

Le paramètre *personal* spécifie le fichier où seront ajoutés les mots du dictionnaire personnel. Ce doit être un chemin absolu, qui commence par '/' car sinon, il sera relatif à \$HOME, qui est '/root' sur la plupart des systèmes, et probablement pas ce que vous souhaitez.

Le paramètre de langage est le code de langue en deux lettres, défini dans la norme ISO 639, et deux lettres optionnelles ISO 3166, après un tiret ou un souligné (_).

Le paramètre d'orthographe *spelling* est nécessaire pour les langues qui ont plus d'une orthographe, comme l'anglais. Les valeurs reconnues sont alors 'american' (américain) , 'british' (anglais), et 'canadian' (canadien). Le paramètre de jargon *jargon* contient des informations supplémentaires pour distinguer deux dictionnaires distincts pour la même langue et le même paramètre d'orthographe *spelling*.

Le paramètre d'encodage indique l'encodage attendu pour la réponse. Les valeurs valides sont : 'utf-8', 'iso8859-*', 'koi8-r', 'viscii', 'cp1252', 'machine unsigned 16', 'machine unsigned 32'. Ce paramètre n'a pas été

testé de manière exhaustive, alors soyez prudent.

Le paramètre de mode est le mode de travail du vérificateur d'orthographe. Plusieurs modes sont disponibles :

- PSPELL_FAST – Mode rapide (moins de suggestions, plus de vitesse)
- PSPELL_NORMAL – Mode normal mode (plus de suggestions)
- PSPELL_BAD_SPELLERS – Mode lent (beaucoup plus de suggestions, moins de vitesse)

Pour plus d'informations et d'exemples, vérifiez le manuel pspell sur leur site web

: <http://pspell.sourceforge.net/>.

Exemple avec pspell_new_personal()

```
<?php
$pspell_link = pspell_new_personal ("/var/dictionaries/custom.pws", "en", "", "", "", PSPELL_FAST
?>
```

10.58.15 pspell_save_wordlist

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_save_wordlist](#)(int *dictionary_link*)

[PHP 4 >= 4.0.2]

[pspell_save_wordlist\(\)](#) sauve le dictionnaire personnel de la session courante. Le dictionnaire doit avoir été ouvert avec [pspell_new_personal\(\)](#), et la localisation des fichiers doit avoir été spécifié avec [pspell_config_personal\(\)](#) et (éventuellement) [pspell_config_repl\(\)](#). Notez que cette fonction n'est pas disponible avec les versions antérieures à pspell .11.2 et aspell .32.5.

Exemple pspell_add_to_personal()

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/tmp/dicts/newdict");
$pspell_link = pspell_new_config ($pspell_config);
pspell_add_to_personal ($pspell_link, "Vlad");
pspell_save_wordlist ($pspell_link);
?>
```

10.58.16 pspell_store_replacement

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [pspell_store_replacement](#)(int *dictionary_link*, string *misspelled*, string *correct*)

[PHP 4 >= 4.0.2]

[pspell_store_replacement\(\)](#) enregistre une paire de remplacement pour un mot de façon à ce que cette suggestion soit retournée par [pspell_suggest\(\)](#) plus tard. Pour pouvoir utiliser cette fonction, vous devez utiliser [pspell_new_personal\(\)](#) pour ouvrir le dictionnaire. Pour pouvoir sauver tout le temps les paires de

remplacement, vous devez utiliser [pspell_config_personal\(\)](#) et [pspell_config_repl\(\)](#) pour indiquer le lieu de sauvegarde des dictionnaires personnels, et [pspell_save_wordlist\(\)](#) pour enregistrer les modifications sur le disque. Ce fichier devra donc être accessible en écriture par PHP. Notez que cette fonction ne fonctionne pas avec les versions antérieures à pspell .11.2 et aspell .32.5.

Exemple avec pspell_store_replacement()

```
<?php
$pspell_config = pspell_config_create ("en");
pspell_config_personal ($pspell_config, "/var/dictionaries/custom.pws");
pspell_config_repl ($pspell_config, "/var/dictionaries/custom.repl");
$pspell_link = pspell_new_config ($pspell_config);
pspell_store_replacement ($pspell_link, $misspelled, $correct);
pspell_save_wordlist ($pspell_link);
?>
```

10.58.17 pspell_suggest

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [pspell_suggest](#) (int *dictionary_link*, string *word*)
[PHP 4 >= 4.0.2]

[pspell_suggest\(\)](#) retourne un tableau de suggestions pour le mot *word*.

pspell_suggest()

```
<?php
$pspell_link = pspell_new ("english");
if (!pspell_check ($pspell_link, "testt")){
    $suggestions = pspell_suggest ($pspell_link, "testt");
    for ($i=0; $i < count ($suggestions); $i++) {
        echo "Orthographes suggerées : " . $suggestions[$i] . "<br>";
    }
}
?>
```

10.59 Readline GNU

[\[Notes en ligne\]](#)

Les fonctions [readline\(\)](#) implémente une interface avec la librairie GNU Readline. Ces fonctions fournissent une ligne de commande éditable, un peu comme lorsque Bash vous permet d'utiliser les flèches de déplacement pour insérer un caractère ou passer en revue l'historique. A cause de l'interactivité de ces commande, elles ne seront que rarement utiles pour les applications Web, mais peuvent se révéler utiles lorsqu'un script est exécuté depuis une commande shell.

Le site du projet GNU Readline est <http://cnswww.cns.cwru.edu/~chet/readline/rltop.html>. Elle est entretenue par Chet Ramey, qui est aussi l'auteur de Bash.

10.59.1 readline

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [readline](#) (string *prompt*)

[PHP 4 >= 4.0b4]

[readline\(\)](#) retourne une ligne entrée par l'utilisateur. Vous pouvez spécifier une chaîne de prompt. La ligne retournée est débarrassée du caractère nouvelle ligne final. Vous devez ajouter cette ligne à l'historique vous-même, avec la fonction [readline_add_history\(\)](#).

Exemple avec readline()

```
<?php
//Lit 3 commandes de l'utilisateur
for ($i=0; $i < 3; $i++) {
    $line = readline("Commande: ");
    readline_add_history($line);
}
//liste l'historique
print_r(readline_list_history());
//liste les variables
print_r(readline_info());
?>
```

10.59.2 readline_add_history

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [readline_add_history](#) (string *line*)

[PHP 4 >= 4.0b4]

[readline_add_history\(\)](#) ajoute une ligne à l'historique.

10.59.3 readline_clear_history

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_clear_history](#) (void)

[PHP 4 >= 4.0b4]

[readline_clear_history\(\)](#) efface tout l'historique.

10.59.4 readline_completion_function

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_completion_function](#) (string *line*)

[PHP 4 >= 4.0b4]

[readline_completion_function\(\)](#) enregistre une nouvelle fonction de complétion. Vous devez fournir le nom d'une fonction qui accepte un nom partiel de commande, et retourne une liste de fonctions complètes possibles. C'est la même fonctionnalité que lorsque vous utilisez la touche de tabulation sous Bash.

10.59.5 readline_info

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [readline_info](#)(string *varname* , string *newvalue*)

[PHP 4 >= 4.0b4]

Appelée sans paramètre, [readline_info\(\)](#) retourne un tableau contenant les valeurs des paramètres de Readline. Les éléments seront indexés par les clés suivantes : done, end, erase_empty_line, library_version, line_buffer, mark, pending_input, point, prompt, readline_name, et terminal_name.

Appelée avec le paramètre *varname*, la valeur de cette variable sera retournée. Appelée avec deux paramètres, et la valeur de la variable *varname*, sera remplacée par *newvalue*.

10.59.6 readline_list_history

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [readline_list_history](#)(void)

[PHP 4 >= 4.0b4]

[readline_list_history\(\)](#) retourne un tableau avec la liste de toutes les lignes de commandes de l'historique. Les éléments sont indexés numériquement, à partir de 0.

10.59.7 readline_read_history

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_read_history](#)(string *filename*)

[PHP 4 >= 4.0b4]

[readline_read_history\(\)](#) lit une ligne de l'historique.

10.59.8 readline_write_history

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [readline_write_history](#)(string *filename*)

[PHP 4 >= 4.0b4]

[readline_write_history\(\)](#) écrit *filename* dans l'historique.

10.60 Recode (GNU)

[\[Notes en ligne\]](#)

Ce module contient l'interface à la librairie GNU Recode library, version 3.5. Pour pouvoir utiliser ces fonctions, il faut que PHP ait été compilé avec l'option `--with-recode`. Pour cela, il faut que vous ayez la librairie GNU Recode 3.5 ou plus récent, installée sur votre système.

La librairie GNU Recode library convertit les fichiers ayant des jeux de caractères différents. Lorsque ce n'est pas possible, elle se débarrasse des caractères illégaux, ou bien effectue une approximation. La librairie reconnaît ou produit près de 150 jeux de caractères différents, et peut quasiment tous les convertir de l'un vers l'autre. La plus part des jeux de caractères de la RFC 1345 sont supportés.

10.60.1 recode_string

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [recode_string](#) (string *request*, string *string*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0RC1]

Recode la chaîne *string* en fonction de la requête *request*. Retourne FALSE, en cas d'échec, et TRUE sinon. Une requête simple de recodage peut être "lat1..iso646-de". Reportez vous à la documentation GNU Recode de votre installation pour plus de détails sur les requêtes.

Exemple simple avec recode_string()

```
print recode_string ("us..flat", "Le caractère suivant est diacritique : &acute;");
```

10.60.2 recode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [recode_string](#) (string *request*, string *string*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0RC1]

Note : [recode_string\(\)](#) est un alias de [recode_string\(\)](#). Elle a été ajoutée dans PHP 4.

10.60.3 recode_file

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [recode_file](#) (string *request*, resource *input*, resource *output*)

[PHP 3>= 3.0.13, PHP 4 >= 4.0RC1]

[recode_file\(\)](#) recode le fichier identifié par *input* dans le fichier identifié par *output* en fonction de la requête de recodage *request*. Retourne FALSE, en cas d'échec, et TRUE sinon.

[recode_file\(\)](#) ne gère pas encore les fichiers distants (URLs). Les deux fichiers doivent faire référence à des fichiers locaux.

Exemple simple avec recode_file()

```
<?php
$input = fopen('input.txt', 'r');
$output = fopen('output.txt', 'w');
recode_file("us..flat", $input, $output);
?>
```

10.61 Expressions régulières

[\[Notes en ligne\]](#)

Les expressions régulières sont utilisées pour effectuer des manipulations complexes de chaînes de

caractères. Les fonctions sont :

- [ereg\(\)](#)
- [ereg_replace\(\)](#)
- [eregi\(\)](#)
- [eregi_replace\(\)](#)
- [split\(\)](#)
- [spliti\(\)](#)

Ces fonctions requièrent toutes une expression régulière comme premier argument. PHP utilise les expressions régulières avancées de POSIX (POSIX 1003.2). Pour avoir tous les détails sur ces expressions, reportez vous aux pages de manuel incluses dans le répertoire de la distribution PHP.

Expressions régulières

```
<?php
ereg("abc",$string);
/* Retourne TRUE si "abc"
est trouvé quelque part dans la chaîne $string. */
ereg("^abc",$string);
/* Retourne TRUE si "abc"
est trouvé au début de la chaîne $string. */
ereg("abc$", $string);
/* Retourne TRUE si "abc"
est trouvé à la fin de la chaîne $string. */
eregi("ozilla.[23]|MSIE.3",$HTTP_USER_AGENT);
/* Retourne TRUE si le client
est Netscape 2, 3 ou MSIE 3. */
ereg("([:alnum:]]+) ([[:alnum:]]+) ([[:alnum:]]+)",
    $string,$regs);
/* Introduit trois mots séparés par des espaces
dans les chaînes $regs[1], $regs[2] et $regs[3]. */
$string = ereg_replace("^","<BR>",$string);
/* Insère une balise <BR> au début de la chaîne $string. */
$string = ereg_replace("$","<BR>",$string);
/* Insère une balise <BR> à la fin de la chaîne $string. */
$string = ereg_replace("\n","", $string);
/* Supprime toutes les nouvelles lignes de $string. */
?>
```

[10.61.1 ereg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ereg](#)(string *pattern*, string *string*, array *regs*)
[PHP 3, PHP 4]

Recherche dans la chaîne *string* les séquences de caractères qui correspondent au masque *pattern*.

Si au moins une séquence est trouvée (éventuellement dans les parenthèses capturantes de *pattern*), et que la fonction est appelée avec un troisième argument *regs*, les résultats seront enregistrés dans *regs*. \$regs[1] contiendra la première parenthèse capturante (celle qui commence le plus tôt), \$regs[2] contiendra la deuxième parenthèse capturante (celle qui commence après la première), et ainsi de suite. \$regs[0] contient une copie de la chaîne.

Si [ereg\(\)](#) trouve ses solutions pour les parenthèses capturantes, *\$regs* contiendra exactement 10 éléments, même si il y avait plus ou moins de 10 parenthèses capturantes qui étaient valides. Cela n'a aucun effet sur les capacités de la fonction [ereg\(\)](#) à trouver d'autres sous chaînes. Si aucune valeur n'est trouvée, *\$regs* ne sera pas modifié par [ereg\(\)](#).

La recherche est sensible à la casse.

[ereg\(\)](#) retourne TRUE si une occurrence a été trouvée dans la chaîne, et FALSE dans le cas contraire, ou si une erreur est survenue.

L'exemple suivant prend une date au format ISO (YYYY-MM-DD) et l'affiche sous la forme DD.MM.YYYY :

Exemple [ereg\(\)](#)

```
<?php
if ( ereg( "([0-9]{4})-([0-9]{1,2})-([0-9]{1,2})", $date, $regs ) ) {
echo "$regs[3].$regs[2].$regs[1]";
} else {
echo "Format de date invalide : $date";
}
?>
```

Voir aussi [ereg\(\)](#), [ereg_replace\(\)](#) et [eregi_replace\(\)](#).

10.61.2 [ereg_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ereg_replace](#)(string *pattern*, string *replacement*, string *string*)
[PHP 3, PHP 4]

Cette fonction effectue une recherche par expression régulière dans la chaîne *string* en recherchant les occurrences de *pattern*, puis les remplace par la chaîne *replacement*.

La chaîne modifiée est retournée. (Ce qui signifie que la chaîne originale sera retournée si aucune occurrence n'est trouvée).

Si *pattern* contient des parenthèses capturantes, *replacement* pourra contenir des séquences de la forme `\\digit`, qui seront remplacées par le texte capturé par la n-ième parenthèse capturante. `\\0` correspond à la chaîne originale complète. De 0 à 9 parenthèses capturantes peuvent être utilisées. Les parenthèses peuvent être imbriquées, et leur numéro d'ordre est défini par leur parenthèse ouvrante.

Si aucune occurrence n'est trouvée, la chaîne *string* sera retournée intacte.

Par exemple, le code suivant affiche "Ceci est un test" trois fois :

Exemple avec [ereg_replace\(\)](#)

```
<?php
$string = "Ceci est un test";
echo ereg_replace( " est", " etait", $string );
echo ereg_replace( "( )est ", "\\1etait", $string );
echo ereg_replace( "(( )est)", "\\2etait", $string );
?>
```

Notez bien que si vous utilisez une valeur de type entier dans le paramètre de remplacement *replacement*, vous risquez de ne pas obtenir le résultat escompté. Tout cela parce que [ereg_replace\(\)](#) va interpréter le nombre comme la valeur ordinaire d'un caractère, et l'utiliser. Par exemple :

Exemple avec [ereg_replace\(\)](#)

```
<?php
/* Cet exemple ne fonctionne pas comme voulu. */
$num = 4;
$string = "Cette chaîne a quatre mots.";
$string = ereg_replace('quatre', $num, $string);
echo $string; /* Affichage : 'Cette chaîne a mots.' */
/* Ceci est bon. */
$num = '4';
$string = "Cette chaîne a quatre mots.";
$string = ereg_replace('quatre', $num, $string);
echo $string; /* Affichage : 'Cette chaîne a 4 mots.' */
?>
```

Voir aussi [ereg\(\)](#), [eregi\(\)](#) et [eregi_replace\(\)](#).

[10.61.3 eregi](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ereg\(\)](#)(string *pattern*, string *string*, array *regs*)

[PHP 3, PHP 4]

[ereg\(\)](#) est identique à [ereg\(\)](#), hormis le fait qu'elle ignore la casse des caractères lors de la recherche sur les caractères alphabétiques.

Voir aussi [ereg\(\)](#), [ereg_replace\(\)](#) et [eregi_replace\(\)](#).

[10.61.4 eregi_replace](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [eregi_replace\(\)](#)(string *pattern*, string *replacement*, string *string*)

[PHP 3, PHP 4]

[eregi_replace\(\)](#) est identique à [ereg_replace\(\)](#), hormis le fait qu'elle ne tient pas compte de la casse des caractères alphabétiques.

Voir aussi [ereg\(\)](#), [eregi\(\)](#) et [ereg_replace\(\)](#).

[10.61.5 split](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [split\(\)](#)(string *pattern*, string *string*, int *limit*)

[PHP 3, PHP 4]

Retourne un tableau de chaînes : chacune d'entre elle est une sous-chaîne de *string* délimitée par les occurrences trouvées de l'expression régulière *pattern*. Si une erreur survient, retourne FALSE.

Pour lire les 5 premiers champs d'une ligne du fichier `/etc/passwd`:

Exemple avec split()

```
<?php
$passwd_list = split( ":", $passwd_line, 5 );
?>
```

Pour analyser une date qui est délimitée par des /, des points ou des tirets :

Exemple avec `split()`

```
<?php
$date = "04/30/1973";
// Les délimiteurs peuvent être des /, des points ou des tirets
list( $month, $day, $year ) = split( '[-./]', $date );
echo "Mois: $month; Jour: $day; Année: $year<br>\n";
?>
```

Notez que ***pattern*** est insensible à la casse

Notez bien que si vous n'avez pas besoin de la puissance des expressions régulières, il est plus rapide d'utiliser [explode\(\)](#), qui n'utilise pas le moteur d'expressions régulières.

Notez aussi que ***pattern*** est une expression régulière. Si vous voulez utiliser n'importe quel caractère spécial des expressions régulières, vous devez les échapper. Si vous pensez que [split\(\)](#) (ou toute autre expression régulière) se comporte bizarrement, lisez d'abord le fichier ``regex.7'`, inclus dans le dossier ``regex/'` de la distribution PHP. Il est au format manpage, et vous pourrez le lire avec une commande telle que `man /usr/local/src/regex/regex.7`.

Voir aussi : [explode\(\)](#) et [implode\(\)](#).

10.61.6 `spliti`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [spliti](#)(string ***pattern***, string ***string***, int ***limit***)
[PHP 3, PHP 4]

[spliti\(\)](#) est identique à [split\(\)](#), hormis le fait qu'elle ignore la casse.

Voir aussi : [split\(\)](#), [explode\(\)](#), et [implode\(\)](#).

10.61.7 `sql_regcase`

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sql_regcase](#)(string ***string***)
[PHP 3, PHP 4]

Retourne une expression régulière valide qui acceptera la chaîne ***string***, et toutes les variantes majuscule/minuscule possibles de cette chaîne. Cette expression sera construite à partir de la chaîne ***string*** en remplaçant tous les caractères par des expressions entre crochets (des classes de caractères), contenant la lettre majuscule et minuscule. Si le caractère n'est pas une lettre, les crochets contiendront deux fois le caractère original.

Exemple avec `sql_regcase()`

```
<?php
echo sql_regcase( "Foo bar" );
?>
```

affichera `[Ff][Oo][Oo] [Bb][Aa][Rr]`.

Cette expression sert à effectuer des recherches insensibles à la casse avec d'autres logiciels, qui n'acceptent les recherches insensibles à la casse.

10.62 Satellite CORBA client extension

[\[Notes en ligne\]](#)

Le module Satellite sert à accéder aux objets CORBA. Vous devez ajouter l'option `idl_directory=` entry dans ``php.ini'` : c'est le chemin jusqu'aux fichiers IDL.

10.62.1 orbitobject

[\[Notes en ligne\]](#)

```
new @xref{function.orbitobject , , orbitobject (string ior) }
```

@xref{function.orbitobject , , orbitobject()} crée un objet qui accèdera aux objets CORBA. Le paramètre *ior* doit être une chaîne contenant l'IOR (Interoperable Object Reference (référence interopérable d'objet)) qui identifie l'objet distant.

Fichier IDL : MonInterface

```
interface MonInterface {
void SetInfo (string info);
string GetInfo();
attribute int value;
}
```

Code PHP pour accéder à MonInterface

```
<?php
$obj = new OrbitObject ($ior);
$obj->SetInfo ("Un Super Objet");
echo $obj->GetInfo();
$obj->value = 42;
echo $obj->value;
?>
```

10.62.2 orbitenum

[\[Notes en ligne\]](#)

```
new @xref{function.orbitenum , , orbitenum (string id) }
```

@xref{function.orbitenum , , orbitenum()} représente l'énumération identifiée par *id*. *id* peut être soit le nom d'une énumération (e.g "MonEnumeration"), ou bien l'identifiant du repository complet (e.g. "IDL:MyEnum:1.0").

Fichier d'exemple IDL : MonEnumeration

```
enum MonEnumeration {
a,b,c,d,e
};
```

Code PHP pour accéder à MonEnumeration

```
<?php
$enum = new OrbitEnum ("MonEnumeration");
echo $enum->a; /* écrit 0 */
echo $enum->c; /* écrit 2 */
echo $enum->e; /* écrit 4 */
?>
```

10.62.3 orbitstruct[\[Notes en ligne\]](#)

`new @xref{function.orbitstruct , , orbitstruct (string id) }`

@xref{function.orbitstruct , , orbitstruct()} représente une structure identifiée par le paramètre *id*. *id* peut être soit le nom d'une structure (e.g "MaStructure"), ou bien l'identifiant du repository complet (e.g. "IDL:MaStructure:1.0").

Fichier d'exemple IDL : MaStructure

```
struct MaStructure {
short shortvalue;
string stringvalue;
};
interface UneInterface {
void SetValues (MaStructure values);
MaStructure GetValues();
}
```

Code PHP pour accéder à MaStructure

```
<?php
$obj = new OrbitObject ($ior);
$initial_values = new OrbitStruct ("IDL:MaStructure:1.0");
$initial_values->shortvalue = 42;
$initial_values->stringvalue = "HGGTG";
$obj->SetValues ($initial_values);
$values = $obj->GetValues();
echo $values->shortvalue;
echo $values->stringvalue;
?>
```

10.62.4 satellite caught exception[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [satellite caught exception](#),

[PHP 4 >= 4.0.3]

[satellite_caught_exception\(\)](#) retourne TRUE si une exception a été émise lors du dernier appel à une fonction Orbit.

Fichier IDL exemple : PlusDeFromage

```
<?php
/* ++?????++ Erreur PlusDeFromage. Recommence tout au début. */
exception PlusDeFromageErreur {
    int parameter;
}
interface UneAutreInterface {
    void AskWhy() raises (PlusDeFromage);
}
?>
```

Code PHP pour gérer les exceptions CORBA

```
<?php
$obj = new OrbitObject ($ior);
$obj->AskWhy();
if (satellite_caught_exception()) {
    if ("IDL:PlusDeFromage:1.0" == satellite_exception_id()) {
        $exception = satellite_exception_value();
        echo $exception->parameter;
    }
}
?>
```

10.62.5 satellite_exception_id

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [satellite_exception_id](#),
[PHP 4 >= 4.0.3]

[satellite_exception_id\(\)](#) retourne une chaîne d'identification d'un repository. (E.g. "IDL:MyException:1.0".)
Pour un exemple, voyez [satellite_caught_exception\(\)](#).

10.62.6 satellite_exception_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

OrbitStruct [satellite_exception_value](#),
[PHP 4 >= 4.0.3]

[satellite_exception_value\(\)](#) retourne une structure d'exception. Pour un exemple, voyez [satellite_caught_exception\(\)](#).

10.63 Sémaphores et gestion de la mémoire partagée

[\[Notes en ligne\]](#)

Ce module fournit un système de sémaphore. Ce système utilise les sémaphores System V. Les sémaphores peuvent être utilisés pour fournir un accès exclusif à certaines ressources de la machine, ou pour limiter le nombre de processus qui utilisent en même temps une ressource.

Ce module fournit aussi un système de mémoire partagée, qui utilise la mémoire partagée System V. Cette mémoire partagée permet d'accéder à des variables globales. Les différents démons httpd et mêmes d'autres programmes (tels que Perl, C, ...) permettent un tel échange de données global. N'oubliez pas que la mémoire partagée n'est pas protégée contre l'accès simultané. Il vous faudra utiliser les sémaphores pour assurer la synchronisation.

SHMMAX	Taille maximale de mémoire partagée, par défaut, 131072 octets. @tab
SHMMIN	Taille minimale de mémoire partagée, par défaut, 1 octet. @tab
SHMMNI	Nombre maximal de segment de mémoire partagé, par défaut 100. @tab
SHMSEG	Taille maximale de mémoire partagée par processus, par défaut 6. @tab

10.63.1 sem_get

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sem_get](#) (int *key*, int *max_acquire* , int *perm*)

[PHP 3>= 3.0.6, PHP 4]

Retourne un identifiant positif de sémaphore en cas de succès, et FALSE en cas d'erreur.

[sem_get\(\)](#) retourne un identifiant qui pourra être utilisé pour accéder à un sémaphore System V. Le sémaphore est créé, si nécessaire, en utilisant les bits de permission (par défaut, 0666). Le nombre de processus qui peuvent réserver simultanément le sémaphore est précisé dans *max_acquire* (par défaut à 1). Actuellement, cette valeur n'est affectée que si le processus est le seul processus actuellement attaché au sémaphore.

Un deuxième appel à [sem_get\(\)](#) avec la même clé retournera un identifiant différent, mais les deux identifiants permettront d'accéder au même sémaphore.

Voir aussi : [sem_acquire\(\)](#) et [sem_release\(\)](#).

Note : Cette fonction n'est pas disponibles sous Windows.

10.63.2 sem_acquire

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sem_acquire](#) (int *sem_identifiant*)

[PHP 3>= 3.0.6, PHP 4]

Retourne TRUE en cas de succès, et FALSE sinon.

[sem_acquire\(\)](#) se bloque (si nécessaire) jusqu'à ce que le sémaphore puisse être réservé. Un processus qui tente de réserver un sémaphore qu'il a déjà réservé restera en attente indéfinie, si cette acquisition excède le nombre *max_acquire* de réservation simultanée.

A la fin d'un script, tous les sémaphores réservés mais non explicitement libérés seront libérés automatiquement, et une alerte sera générée.

Voir aussi : [sem_get\(\)](#) et [sem_release\(\)](#).

10.63.3 sem_release

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sem_release](#)(int *sem_identifieur*)

[PHP 3>= 3.0.6, PHP 4]

Retourne TRUE en cas de succès, FALSE en cas d'erreur.

[sem_release\(\)](#) libère le sémaphore s'il a été réservé par le processus courant. Sinon, génère une erreur.

Après libération du sémaphore, [sem_acquire\(\)](#) peut être appelé pour le réserver à nouveau.

Voir aussi : [sem_get\(\)](#) et [sem_acquire\(\)](#).

Note : *Cette fonction n'est pas disponibles sous Windows.*

10.63.4 shm_attach

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_attach](#)(int *key*, int *memsize* , int *perm*)

[PHP 3>= 3.0.6, PHP 4]

[shm_attach\(\)](#) retourne un identifiant qui permettra d'accéder au System V de mémoire partagée. Au premier appel, la mémoire sera créée, avec la taille *mem_size* (par défaut: `sysvshm.init_mem` dans ``php3.ini'`, sinon 10000 octets) et avec les permissions *perm*(par défaut : 666).

Aux appels suivants avec la même clé *key*, [shm_attach\(\)](#) retournera un nouvel identifiant, mais cet identifiant accèdera toujours à la même portion de mémoire partagée. Dans ce cas, *memsize* et *perm* seront ignorés.

Note : *Cette fonction n'est pas disponibles sous Windows.*

10.63.5 shm_detach

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_detach](#)(int *shm_identifieur*)

[PHP 3>= 3.0.6, PHP 4]

[shm_detach\(\)](#) rel'che le segment de mémoire partagée identifié par *shm_identifieur* et créé par [sem_get\(\)](#).

N'oubliez pas que cette mémoire partagée existe toujours sous Unix, et que les données sont toujours accessibles.

10.63.6 shm_remove

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_remove](#)(int *shm_identifieur*)

[PHP 3>= 3.0.6, PHP 4]

Supprime un segment de mémoire partagée sous Unix. Toutes les données seront supprimées.

Note : *Cette fonction n'est pas disponibles sous Windows.*

10.63.7 shm_put_var

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_put_var](#) (int *shm_identifieur*, int *variable_key*, mixed *variable*)

[PHP 3>= 3.0.6, PHP 4]

Insère ou modifie la variable *variable* avec la clé *variable_key*. Tous les types de variables (double, int, string, array) sont supportés.

Note : Cette fonction n'est pas disponibles sous Windows.

10.63.8 shm_get_var

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [shm_get_var](#) (int *id*, int *variable_key*)

[PHP 3>= 3.0.6, PHP 4]

[shm_get_var\(\)](#) retourne la variable repérée par *variable_key*. La variable est toujours présente en mémoire partagée.

Note : Cette fonction n'est pas disponibles sous Windows.

10.63.9 shm_remove_var

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_remove_var](#) (int *id*, int *variable_key*)

[PHP 3>= 3.0.6, PHP 4]

[shm_remove_var\(\)](#) efface la variable *variable_key* de la mémoire partagée et libère la mémoire.

Note : Cette fonction n'est pas disponibles sous Windows.

10.64 SESAM

[\[Notes en ligne\]](#)

SESAM/SQL-Server est une base de données mainframe, développée par Fujitsu Siemens Computers, Allemagne. Elle fonctionne sur les serveur mainframe, sous BS2000/OSD.

Sur de nombreuses installation BS2000 en production, SESAM/SQL-Server a prouvé ...

- La facilité de connectivité Java, Web et client/serveur
- La disponibilité de plus de 99.99%,
- La capacité de gérer des dizaines et mêmes des centaines de milliers d'utilisateurs.

Désormais, il existe une interface PHP pour SESAM, qui donne l'accès à cette base aux scripts PHP.

Note : Il n'y a pas de support pour l'interface SESAM si PHP est un CGI : elle ne fonctionne que comme module Apache. En module Apache, l'interface [7.1.9 Directives de configuration SESAM](#) peut être configurée avec les directives Apache.

Directive	Signification
-----------	---------------

`php3_sesam_oml` @tab Nom de la librairie BS2000 PLAM contenant le module du pilote SESAM. Ceci est obligatoire pour utiliser les fonctions SESAM. Exemple:
`php3_sesam_oml $.SYSLNK.SESAM-SQL.030`

@tab

`php3_sesam_configfile` @tab Nom du fichier de configuration de l'application SESAM. Ceci est obligatoire pour utiliser les fonctions SESAM. Exemple:
`php3_sesam_configfile $SESAM.SESAM.CONF.AW`

Ce fichier contient généralement une configuration comme celle ci (voir dans le manuel de référence SESAM):

```
CNF=B
NAM=K
NOTYPE
```

@tab

`php3_sesam_messagecatalog` @tab Nom du fichier de messages SESAM. Dans la plus part des cas, cette directive n'est pas nécessaire. Uniquement si le fichier de messages SESAM n'es pas installé dans la table de messages BS2000. Il peut alors être choisi avec cette directive. Exemple:
`php3_sesam_messagecatalog $.SYSMES.SESAM-SQL.030`

@tab

En plus de la configuration de l'interface PHP/SESAM, vous devez aussi configurer le serveur SESAM-Database sur votre mainframe, comme d'habitude. Cela signifie notamment qu'il faut :

- démarrer le gestionnaire de base SESAM (DBH)
- connecter les bases avec le gestionnaire de bases SESAM

Pour connecter un script PHP au serveur de bases SESAM, les paramètres CNF et NAM de la configuration SESAM sélectionnée doivent correspondre à l'id du gestionnaire de base démarré.

Dans le cas des bases de données distribuées, vous devez démarrer un agent SESAM/SQL-DCN, avec la table de distribution incluant le nom de l'hôte et de la base de données.

La communication entre PHP (fonctionnant sur le sous-système POSIX) et le gestionnaire de base (fonctionnant hors du sous-système POSIX) est réalisée par un pilote spécial appelé SQLSCI et le module de connexion SESAM, qui utilise la mémoire partagée. A cause de la mémoire partagée, et parce que PHP est une partie statique du serveur web, les accès à la base de données sont extrêmement rapide, car il ne requièrent pas de connexion distante via ODBC, JDBC ou UTM.

Seul un chargeur de stub (stub loader, SESMOD) est compilé dans PHP. Les modules de connexion SESAM proviennent de la librairie OML PLAM. Dans la [7.1.9 Directives de configuration SESAM](#), vous devez indiquer à PHP le nom de la librairie PALM, et le fichier de lien à utiliser pour la configuration de SESAM

(En SESAM V3.0, SQLSCI est disponible dans la librairie d'outils SESAM (SESAM Tool Library), qui fait partie de la distribution standard).

Les commandes SQL imposent que les guillemets simples soient doublés pour être interprété littéralement (contrairement à d'autres bases de données qui utilisent un guillemet simple, précédé d'un antislash), il est recommandé d'activer les directives PHP [7.1.1.17 ini.magic-quotes-gpc](#) et [7.1.1.19 ini.magic-quotes-sybase](#).

Note : A cause des limitations du modèle de processus BS2000, le pilote peut être chargé uniquement après que le serveur Apache ait généré le processus fils. Cela ralentit légèrement le traitement de la première requête, mais toutes les requêtes suivantes seront effectuée à pleine vitesse.

Lorsque vous définissez explicitement le catalogue de messages SESAM, ce catalogue sera chargé à chaque fois que le pilote est chargé (i.e., au moment de la requête initiale). Le système d'exploitation BS2000 affiche un message après avoir correctement chargé le catalogue de messages, qui sera envoyé au fichier d'erreurs Apache. BS2000 ne permet pas la suppression de ce message, qui va remplir progressivement ce fichier. Assurez vous que la librairie SESAM OML PLAM et le fichier de configuration SESAM sont accessibles par l'utilisateur qui fait tourner le serveur web. Sinon, le serveur ne sera pas capable de charger le pilote, ou d'appeler les fonctions SESAM. L'accès à la base doit être donné à cet utilisateur. Sinon, les connexions SESAM échoueront.

Note : Les curseurs de résultat sont alloués pour les requêtes SQL de sélection, peuvent être soit "séquentiels", soit "à défilement" ("scrollable"). Les curseurs à défilement sont beaucoup plus gourmands en mémoire, et le mode par défaut est séquentiel.

Lorsque vous utilisez les curseurs à défilement, le curseur peut être positionné librement dans le résultat. Pour chaque requête à défilement, il existe des valeurs globales de types de défilement (initialisée à :SESAM_SEEK_NEXT) et la position peut être fixée une seule fois par [sesam_seek_row\(\)](#) ou bien à chaque appel, avec la fonction [sesam_fetch_row\(\)](#). Lorsque vous lisez une ligne avec un curseur à défilement, le traitement suivant est effectué à partir des valeurs globales de type de défilement et de position :

Type de défilement	Action
SESAM_SEEK_NEXT @tab aucun	
SESAM_SEEK_PRIOR @tab aucun	
SESAM_SEEK_FIRST @tab le type de défilement devient SESAM_SEEK_NEXT @tab	
SESAM_SEEK_LAST @tab le type de défilement devient SESAM_SEEK_PRIOR @tab	
SESAM_SEEK_ABSOLUTE @tab incrémente automatiquement la valeur interne de position @tab	
SESAM_SEEK_RELATIVE @tab aucune. conserve les valeurs globales par défaut de position, ce qui permet, par exemple de lire toutes les 10 lignes, en arrière. @tab	

Note : En PHP, il est naturel de commencer les index à zéro (plutôt que 1), et quelques adaptations ont été faites pour l'interface SESAM : à chaque fois qu'un tableau indexé commence à l'index 1 en SESAM natif, l'interface PHP utilisera l'index 0 comme point de départ. Par exemple, lorsque vous lisez des données avec [sesam_fetch_row\(\)](#), la première colonne sera à l'index 0, et les suivantes suivront jusqu'au nombre de colonne (exclus) du résultat (\$array["count"]). Lors du portage d'applications depuis d'autres langages évolués vers le PHP, soyez attentifs à ce changement. A chaque fois que c'est nécessaire, la description d'une fonction PHP SESAM indique que l'index du tableau commence à 0.

Note : Lorsque vous autorisez l'accès à une base de données SESAM, le serveur web doit avoir le minimum de privilèges possible. Pour la plus part des bases de données, seul le droit de lecture doit être fourni. Suivant votre utilisation, ajoutez d'autres droits d'accès au fur et à mesure de vos besoins. Ne donnez jamais le contrôle total de vos bases à un utilisateur du web! Limitez l'accès aux scripts PHP qui doivent administrer la base en utilisant un mot de passe et/ou une sécurisation SSL.

Note : Deux langage SQL ne sont jamais 100% compatibles. Lorsque vous portez une application SQL depuis une autre interface vers SESAM, certaines adaptation doivent être faites. Les différences suivantes sont les plus courantes :

- *Types de données spécifiques*
Certains types de données spécifiques à une base doivent être remplacés par les types de données standard SQL. (i.e., TEXT doit être remplacé par VARCHAR(taille max)).
- *Mots réservés comme identifiants SQL.*
En SESAM (comme dans le standard SQL), les mots réservés utilisés comme identifiants doivent être entourés de guillemets doubles (ou renommés).
- *Taille d'affichage des données.*
Les types de données SESAM ont une taille de stockage, mais par de taille d'affichage. A la place de int(4) (c'est à dire : les entiers jusqu'à '9999'), SESAM requiert simplement int, pour une taille implicite de 31 bits. De même, les seuls types de date disponible dans SESAM sont : DATE, TIME(3) et TIMESTAMP(3).
- *Les types de données unsigned (non signé), zerofill (complété avec des zéros), ou auto_increment*
Unsigned et zerofill ne sont pas supportés. Auto_increment est automatique (utilisez "INSERT ... VALUES(*, ...)" au lieu de "... VALUES(0,...)" pour profiter des auto-increment implicites de SESAM.
- *int ... DEFAULT '0000'*
Les variables numériques ne doivent pas être initialisées avec des constantes de type chaîne de caractères. Utilisez DEFAULT 0 à la place. Pour initialiser une date, la chaîne doit être préfixée avec le type de date adapté, tel que : CREATE TABLE exmpl (xtime timestamp(3) DEFAULT TIMESTAMP '1970-01-01 00:00:00.000' NOT NULL);
- *\$count = xxxx_num_rows();*
Certaines bases de données essaient d'estimer le nombre de lignes d'un résultat, même grossièrement approximativement. SESAM ne connaît pas le nombre de lignes avant de les avoir lues lui-même. Si vous avez vraiment besoin de les compter, utilisez la commande SELECT COUNT(. . .) WHERE . . . , qui vous dira combien de lignes sont disponibles. Une deuxième requête devrait vous retourner tous ces résultats.
- *DROP TABLE lenom;*
Avec SESAM, dans la commande DROP TABLE, le nom de la table doit être suivi du mot clé RESTRICT ou CASCADE. Avec RESTRICT, une erreur est retournée si il y a des objets dépendant (par exemple, des vues), tandis qu'avec CASCADE, les objets dépendants seront supprimés en même temps que la table.

Note : SESAM ne supporte pas le type BLOB. Une future version de SESAM devra le faire.
L'interface PHP effectue automatiquement les conversions suivantes lors de la lecture de lignes de résultats SQL :

Type SQL	Type PHP
SMALLINT, INTEGER	"integer" (entier)
NUMERIC, DECIMAL, FLOAT, REAL, DOUBLE	"double" (nombre à virgule flottante)
DATE, TIME, TIMESTAMP	"string"(chaîne de caractères)
VARCHAR, CHARACTER	"string"(chaîne de caractères)

Lorsque vous lisez une ligne entière, le résultat est retourné sous la forme d'un tableau. Les champs vides ne

sont pas remplis, et vous aurez à vérifier vous même l'existence des champs (utilisez [isset\(\)](#) ou [empty\(\)](#) pour tester les champs vides). Cela donne plus de contrôle à l'utilisateur sur l'apparence des champs que si les champs vides étaient représenté par des chaînes vides).

Note : La fonctionnalité spéciale des "champs multiples" de SESAM permet à une colonne de contenir un tableau de champs. Un tel "champs multiple" peut être créé comme ceci :

Création d'une colonne de champs multiples

```
CREATE TABLE multi_field_test
(
  pkey CHAR(20) PRIMARY KEY,
  multi(3) CHAR(12)
)
```

et peut être remplie avec :

Affectation d'une colonne de type "champs multiple"

```
INSERT INTO multi_field_test ( pkey, multi(2..3) )
VALUES ( 'Second', <'first_val','second_val'>)
```

Notez que (comme c'est le cas ci-dessus), les sous-champs vides initiaux sont ignorés, et que le tableau est alors compacté, ce qui fait que l'exemple ci-dessus conduit à un tableau multi(1..2) au lieu de multi(2..3). Lors de la lecture d'une ligne, les "champs multiples" sont mis en colonne. Dans l'exemple ci-dessus, "pkey" prend l'index 0, et les trois colonnes "multi(1..3)" sont accessibles depuis les index 1 à 3.

Pour de plus amples détails sur SESAM, reportez vous à la documentation [SESAM/SQL-Server](#) en anglais ou [SESAM/SQL-Server](#) en allemand, disponibles toutes deux en ligne, ou en manuels.

10.64.1 sesam_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [sesam_connect](#)(string *catalog*, string *schema*, string *user*)

[sesam_connect\(\)](#) retourne TRUE si une connexion à la base SESAM a été faite, ou FALSE en cas d'erreur.

[sesam_connect\(\)](#) établit une connexion au serveur SESAM. La connexion est toujours "persistante", en ce sens que le pilote sera chargé par la première requête avec la librairie SESAM OML PLAM. Les appels suivants réutiliseront le pilote chargé, son catalogue *catalog*, son schéma *schema* et son utilisateur *user*.

Lors de la création d'une base de données, le nom *catalog* est spécifié dans les directives de configuration SESAM avec la commande //ADD-SQL-DATABASE-CATALOG-LIST ENTRY-1 =

```
*CATALOG(CATALOG-NAME = catalogname, ...)
```

schema référence le schéma de base voulu (voir dans le manuel SESAM).

user spécifie l'un des utilisateurs qui est autorisé à accéder à la combinaison *catalog* et/ou *schema*. Notez que *user* est complètement indépendant de l'utilisateur système et des protections HTTP par mot de passe. Il n'apparaît que dans la configuration SESAM.

Voir aussi [sesam_disconnect\(\)](#).

Connexion à une base SESAM

```
<?php
if (! sesam_connect ("moncatalogue", "monschema", "toto")
die("Impossible de se connecter à SESAM");
?>
```


10.64.2 [sesam_disconnect](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [sesam_disconnect](#)

[sesam_disconnect\(\)](#) retourne toujours TRUE.

[sesam_disconnect\(\)](#) ferme le lien logique à la base de données SESAM (sans réellement déconnecter et démonter le pilote).

Notez que ceci n'est généralement pas nécessaire, car la connexion ouverte est automatiquement fermée à la fin du script. Les données qui ne seront pas validées seront alors annulées, grâce à un

[sesam_rollback\(\)](#) implicite.

[sesam_disconnect\(\)](#) ne ferme pas les connexions persistantes : elle invalide simplement les catalogues *catalog*, schéma *schema* et utilisateur *user* courants, de manière à ce que les prochains appels à des fonctions SESAM échouent.

Voir aussi : [sesam_connect\(\)](#).

Déconnexion d'une base SESAM

```
<?php
if (sesam_connect ("moncatalogue", "monschema", "toto")) {
... quelques requêtes et d'autres trucs ...
sesam_disconnect();
}
?>
```

10.64.3 [sesam_settransaction](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [sesam_settransaction](#) (int *isolation_level*, int *read_only*)

[sesam_settransaction\(\)](#) retourne TRUE si les valeurs sont valides et que la modification a été réussie. FALSE sinon.

[sesam_settransaction\(\)](#) remplace les valeurs par défaut du niveau d'isolation ("isolation level") et de lecture seule ("read-only") fixée par le fichier de configuration SESAM), afin d'optimiser les requêtes ultérieures et garantir la cohérence de la base. Ces valeurs ne sont utilisées que pour la prochaine transaction.

[sesam_settransaction\(\)](#) ne peut être appelée qu'avant le début de la transaction. Elle est inefficace si la transaction a déjà commencé.

Pour simplifier l'utilisation de cette fonction dans les scripts PHP, les constantes suivantes ont été définies en PHP (reportez vous au manuel SESAM pour avoir des détails sur leur signification) :

Valeur	Constante	Signification
1	SESAM_TXISOL_READ_UNCOMMITTED	Lecture sans validation
2	SESAM_TXISOL_READ_COMMITTED	Lecture avec validation
3	SESAM_TXISOL_REPEATABLE_READ	Lecture récurente
4	SESAM_TXISOL_SERIALIZABLE	Sérialisable

Valeur	Constante	Signification
--------	-----------	---------------

0	SESAM_TXREAD_READWRITE	Lecture/écriture
1	SESAM_TXREAD_READONLY	Lecture seule

Les valeurs modifiées par [sesam_settransaction\(\)](#) remplaceront les valeurs par défaut spécifiée dans [7.1.9.2 ini.sesam-configfile](#).

Modifier les paramètres de configuration SESAM

```
<?php
sesam_settransaction(SESAM_TXISOL_REPEATABLE_READ,
SESAM_TXREAD_READONLY);
?>
```

10.64.4 sesam_commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [sesam_commit](#)

[sesam_commit\(\)](#) retourne TRUE en cas de succès et FALSE sinon.

[sesam_commit\(\)](#) valide toutes les modifications de tables en attente sur la base.

Notez qu'il n'y a pas de mode "auto-commit", comme dans d'autres bases de données, car cela peut conduire à une perte accidentelle de données. Les données non valides à la fin d'un script (ou au moment de l'appel de [sesam_disconnect\(\)](#)) seront annulées par un appel implicite à [sesam_rollback\(\)](#).

Voir aussi : [sesam_rollback\(\)](#).

Valider une transaction SESAM

```
<?php
if (sesam_connect ("moncatalogue", "monschema", "toto")) {
if (!sesam_execimm("INSERT INTO mytable VALUES (*, 'Small Test', <0, 8, 15>))")
die("insertion manquée");
if (!sesam_commit())
die("insertion réussie");
}
?>
```

10.64.5 sesam_rollback

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [sesam_rollback](#)

[sesam_rollback\(\)](#) retourne TRUE en cas de succès et FALSE en cas d'erreur.

[sesam_rollback\(\)](#) annule toutes les modifications en cours sur la base. Les curseurs de résultat et les descripteurs de résultats seront affectés.

A la fin de chaque script, et dans chaque appel à [sesam_disconnect\(\)](#), un appel implicite à [sesam_rollback\(\)](#) est fait, annulant toutes les transactions non validées dans la base.

Voir aussi : [sesam_commit\(\)](#).

Annulation d'une transaction SESAM

```
<?php
if (sesam_connect ("moncatalogue", "monschema", "toto")) {
if (sesam_execimm("INSERT INTO matable VALUES (*, 'Petit Test', <0, 8, 15>))
    38;sesam_execimm("INSERT INTO autretable VALUES (*, 'Autre Test', 1)"))
    sesam_commit();
else
    sesam_rollback();
}
?>
```

10.64.6 sesam_execimm

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sesam_execimm](#) (string *query*)

[sesam_execimm\(\)](#) retourne un identifiant de résultat SESAM en cas de succès, et FALSE sinon.

[sesam_execimm\(\)](#) exécute immédiatement la requête *query* (i.e., une requête de type UPDATE, INSERT ou DELETE qui ne retourne aucun résultat, et n'a aucune variables d'entrées ou de sorties). Les requêtes de types "SELECT" ne peuvent pas être utilisées avec la fonction [sesam_execimm\(\)](#). [sesam_execimm\(\)](#) modifie la valeur *affected_rows*, pour lecture ultérieure avec [sesam_affected_rows\(\)](#).

Notez que [sesam_query\(\)](#) peut gérer les requêtes immédiates et les requêtes de selection. Utilisez [sesam_execimm\(\)](#) uniquement si vous connaissez le type de requête auparavant. Une tentative de requête de selection avec [sesam_execimm\(\)](#) retournera \$err["sqlstate"] == "42SBW".

L'identifiant de résultat retourné ne peut pas être utilisé pour lire quoi que ce soit, mais il peut être passé à [sesam_affected_rows\(\)](#); il n'est retourné que pour symétrie avec la fonction [sesam_query\(\)](#).

```
<?php
$stmt = "INSERT INTO matable VALUES('un', 'deux')";
$result = sesam_execimm ($stmt);
$error = sesam_diagnostic();
print("sqlstate = ".$error["sqlstate"]."\n".
      "Nombre de lignes affectées = ".$error["rowcount"]." == ".
    sesam_affected_rows($result)."\n");
?>
```

Voir aussi : [sesam_query\(\)](#) et [sesam_affected_rows\(\)](#).

10.64.7 sesam_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sesam_query](#) (string *query*, boolean *scrollable*)

[sesam_query\(\)](#) retourne un identifiant de résultat SESAM en cas de succès, ou FALSE en cas d'erreur.

L'identifiant de résultat est utilisé par d'autres fonctions sesam pour lire les valeurs.

[sesam_query\(\)](#) envoie une requête à la base active. Elle peut exécuter aussi bien une requête immédiate (DELETE, UPDATE ou INSERT), ou une requête de selection. Si une requête immédiate est exécutée, aucun curseur n'est alloué, et il ne sera pas possible d'utiliser les fonctions [sesam_fetch_row\(\)](#) ou

[sesam_fetch_result\(\)](#). Pour les requêtes de sélection, un descripteur de résultat et un curseur (scrollable ou séquentiel, suivant le paramètre optionnel *scrollable* passé) sear alloué. Si *scrollable* est omis, le curseur sera séquentiel.

Lorsque vous utilisez les curseurs à défilement, le curseur peut être positionné librement dans le résultat. Pour chaque requête à défilement, il existe des valeurs globales de types de défilement (initialisée à :SESAM_SEEK_NEXT) et la position peut être fixée une seule fois par [sesam_seek_row\(\)](#) ou bien à chaque appel, avec la fonction [sesam_fetch_row\(\)](#).

Pour les requêtes immédiates, le nombre de lignes affectées est sauvé, et est accessible par la fonction [sesam_affected_rows\(\)](#).

Voir aussi : [sesam_fetch_row\(\)](#) et [sesam_fetch_result\(\)](#).

Liste toutes les lignes de table "phone" sous forme de table HTML

```
<?php
if (!sesam_connect("phonedb", "demo", "toto"))
die("cannot connect");
$result = sesam_query("select * from phone");
if (!$result) {
    $err = sesam_diagnostic();
    die($err["errmsg"]);
}
echo "<TABLE BORDER>\n";
// Ajoute l'entête de titre comme nom de colonne
if ($cols = sesam_field_array($result)) {
echo " <TR><TH COLSPAN=". $cols["count"].>Result:</TH></TR>\n";
echo " <TR>\n";
for ($col = 0; $col < $cols["count"]; ++$col) {
    $colattr = $cols[$col];
    /* étend les entêtes de la table au dessus des champs multiples */
    if ($colattr["count"] > 1) {
echo " <TH COLSPAN=". $colattr["count"].">". $colattr["name"].
        "(1..". $colattr["count"].")</TH>\n";
        $col += $colattr["count"] - 1;
    }
    else
echo " <TH>". $colattr["name"] . "</TH>\n";
    }
echo " </TR>\n";
}
do {
    // Lit les résultats par bloc de 100
    $ok = sesam_fetch_result($result,100);
    for ($row=0; $row < $ok["rows"]; ++$row) {
echo " <TR>\n";
        for ($col = 0; $col < $ok["cols"]; ++$col) {
            if (isset($ok[$col][$row]))
echo " <TD>". $ok[$col][$row] . "</TD>\n";
            else
echo " <TD>-empty-</TD>\n";
        }
echo " </TR>\n";
    }
} while ($ok["truncated"]); // tant qu'il a y encore des données
echo "</TABLE>\n";
// libère les ressources
sesam_free_result($result);
?>
```

10.64.8 [sesam_num_fields](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sesam_num_fields](#)(string *result_id*)

Après avoir appelé [sesam_query\(\)](#) avec une requête de sélection, [sesam_num_fields\(\)](#) indique le nombre de colonnes du résultat identifié par *result_id*. Retourne FALSE en cas d'erreur.

Pour les requêtes immédiates, la valeur zéro est retournée. Les champs multiples SESAM compte autant que leur taille respective, c'est à dire qu'un champs multiple de trois colonnes compte comme trois colonne.

Voir aussi : [sesam_query\(\)](#) et [sesam_field_array\(\)](#) pour savoir distinguer les champs multiples des colonnes standard.

10.64.9 [sesam_field_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sesam_field_name](#)(string *result_id*, int *index*)

[sesam_field_name\(\)](#) retourne le nom du champs *index* dans le résultat identifié par *result_id*, ou FALSE en cas d'erreur.

Pour les requêtes immédiates, ou les colonnes dynamiques, une chaîne vide est retournée.

Note : *Les colonnes sont indexées à partir de 0, et non pas 1.*

Voir aussi : [sesam_field_array\(\)](#). Cette fonction fournit une interface simple aux noms et types de colonnes, et permet la detection des champs multiples.

10.64.10 [sesam_diagnostic](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sesam_diagnostic](#)

[sesam_diagnostic\(\)](#) retourne un tableau associatif avec l'état et les codes de la dernière requête SQL. Les éléments du tableau sont :

Élément	Contenu
<code>\$array["sqlstate"]</code>	code d'erreur à 5 chiffres (voir le manuel SESAM pour obtenir une description des valeurs possibles de SQLSTATE) @tab
<code>\$array["rowcount"]</code>	nombre de lignes affectées dans la dernière requête immédiate (update/insert/delete) : uniquement après une requête immédiate. @tab
<code>\$array["errmsg"]</code>	message d'erreur lisible : uniquement après une erreur @tab
<code>\$array["errcol"]</code>	numéro de colonne de la dernière erreur (indexée à partir de 0, -1 si indéfinies. uniquement après une erreur). @tab
<code>\$array["errlin"]</code>	numéro de ligne de la dernière erreur

(indexée à partir de 0, -1 si indéfinies.
uniquement après une erreur). @tab

Dans l'exemple suivant, une erreur de syntaxe (E SEW42AE ILLEGAL CHARACTER) est affichée avec la requête SQL, et en désignant la position de l'erreur :

Afficher une erreur SESAM

```
<?php
// Fonction qui affiche un message d'erreur formaté
// en affichant la position de l'erreur dans le message d'erreur
function PrintReturncode($exec_str)
{
    $err = Sesam_Diagnostic();
    $colspan=4; // 4 colonnes pour : sqlstate, errlin, errcol, rowcount
    if ($err["errlin"] == -1)
        --$colspan;
    if ($err["errcol"] == -1)
        --$colspan;
    if ($err["rowcount"] == 0)
        --$colspan;
    echo "<TABLE BORDER>\n";
    echo "<TR><TH COLSPAN=".$colspan."><FONT COLOR=red>ERROR:</FONT> ".
    htmlspecialchars($err["errmsg"])."</TH></TR>\n";
    if ($err["errcol"] >= 0) {
        echo "<TR><TD COLSPAN=".$colspan."><PRE>\n";
        $errstmt = $exec_str."\n";
        for ($lin=0; $errstmt != ""; ++$lin) {
            if ($lin != $err["errlin"]) { // $lin is less or greater than errlin
                if (! ($i = strchr($errstmt, "\n")))
                    $i = "";
                $line = substr($errstmt, 0, strlen($errstmt)-strlen($i)+1);
                $errstmt = substr($i, 1);
                if ($line != "\n")
                    print htmlspecialchars($line);
            }
            else {
                if (! ($i = strchr($errstmt, "\n")))
                    $i = "";
                $line = substr($errstmt, 0, strlen($errstmt)-strlen($i)+1);
                $errstmt = substr($i, 1);
                for ($col=0; $col < $err["errcol"]; ++$col)
                    echo (substr($line, $col, 1) == "\t") ? "\t" : ".";
                echo "<FONT COLOR=RED><BLINK>\\</BLINK></FONT>\n";
                print "<FONT COLOR=\\\"#880000\\\">".htmlspecialchars($line)."</FONT>";
                for ($col=0; $col < $err["errcol"]; ++$col)
                    echo (substr($line, $col, 1) == "\t") ? "\t" : ".";
                echo "<FONT COLOR=RED><BLINK></BLINK></FONT>\n";
            }
        }
        echo "</PRE></TD></TR>\n";
    }
    echo "<TR>\n";
    echo " <TD>sqlstate=" . $err["sqlstate"] . "</TD>\n";
    if ($err["errlin"] != -1)
        echo " <TD>errlin=" . $err["errlin"] . "</TD>\n";
    if ($err["errcol"] != -1)
        echo " <TD>errcol=" . $err["errcol"] . "</TD>\n";
    if ($err["rowcount"] != 0)
        echo " <TD>rowcount=" . $err["rowcount"] . "</TD>\n";
}
```

```

echo "</TR>\n";
echo "</TABLE>\n";
}
if (!sesam_connect("moncatalogue", "phoneno", "toto"))
die("cannot connect");
$stmt = "SELECT * FROM phone\n".
        "  WHERE LASTNAME='KRAEMER'\n".
        "  ORDER BY FIRSTNAME";
if (! ($result = sesam_query($stmt)))
PrintReturncode($stmt);
?>

```

Voir aussi : [sesam_errormsg\(\)](#) pour un accès simplifié aux messages d'erreur.

10.64.11 [sesam_fetch_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [sesam_fetch_result](#) (string *result_id* , int *max_rows*)

[sesam_fetch_result\(\)](#) retourne un tableau avec les lignes du résultat identifié par *result_id*, éventuellement limité à un maximum de *max_rows*. Notez que les lignes et les colonnes sont indexées à partir de 0.

Élément du tableau	Contents
int \$arr["count"]	Nombre de colonnes dans le résultat (ou zéro si c'était une requête immédiate). @tab
int \$arr["rows"]	Nombre de ligne dans le résultat (entre zéro et <i>max_rows</i>) @tab
boolean \$arr["truncated"]	TRUE si le nombre de ligne était d'au moins <i>max_rows</i> , FALSE sinon. Notez que même si cette valeur est à TRUE, le prochain appel à sesam_fetch_result() peut retourner aucune ligne parce qu'il n'y a plus d'entrées. @tab
mixed \$arr[col][row]	les valeurs du résultat à la ligne row et colonne col. Le résultat est un tableau multidimensionnel. row va de 0 à \$arr["rows"]-1, et col de 0 à \$arr["count"]-1). Les champs peuvent être vides : vous devez vérifier leur existence avec la fonction isset() . Le type retourné dépend du type SQL déclaré pour cette colonne (voir 10.64 SESAM pour connaître les conversions utilisées). Les champs multiples SESAM sont traités comme des séquences de colonnes. @tab

Notez que la quantité de mémoire utilisée par des requêtes peut se révéler gigantesque. Utilisez alors *max_rows* pour limiter le nombre maximum de lignes retournées, à moins que vous ne soyez absolument sûr

que votre résultat ne consommera pas toute la mémoire disponible.

Voir aussi : [sesam_fetch_row\(\)](#), et [sesam_field_array\(\)](#) pour vérifier les champs multiples. Voyez [sesam_query\(\)](#) pour un exemple complet avec [sesam_fetch_result\(\)](#).

10.64.12 [sesam_affected_rows](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sesam_affected_rows](#)(string *result_id*)

result_id est un identifiant valide de résultat, retourné par [sesam_query\(\)](#).

[sesam_affected_rows\(\)](#) retourne le nombre de lignes affectées par la requête associée à *result_id*.

[sesam_affected_rows\(\)](#) ne retourne de valeur cohérente que lorsqu'utilisée avec une requête immédiate (INSERT/UPDATE/DELETE), car SESAM ne fournit aucune information de nombre de lignes affectées pour les requêtes de sélection.

Voir aussi : [sesam_query\(\)](#) et [sesam_execimm\(\)](#)

```
<?php
$result = sesam_execimm ("DELETE FROM PHONE WHERE LASTNAME = '".strtoupper($name)."'");
if (! $result) {
    ... error ...
}
print sesam_affected_rows($result). " entries with last name ".$name." deleted.\n";
?>
```

10.64.13 [sesam_errormsg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sesam_errormsg](#)

[sesam_errormsg\(\)](#) retourne le message d'erreur SESAM associé à la dernière requête SQL.

```
<?php
if (!sesam_execimm($stmt))
printf("%s<br>\n", sesam_errormsg());
?>
```

Voir aussi : [sesam_diagnostic\(\)](#) pour la liste complète des états de requêtes SQL.

10.64.14 [sesam_field_array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sesam_field_array](#)(string *result_id*)

result_id is a valid result id returned by [sesam_query\(\)](#).

[sesam_field_array\(\)](#) retourne un tableau contenant les informations (nom de colonne, type, précision...) sur une colonne dans le résultat associé à *result_id*.

Index	Contenu
int \$arr["count"]	Nombre total de colonnes dans le résultat (ou zéro si la requête était immédiate). Les champs multiples de SESAM sont linéarisés, et traités comme

	autant de colonnes. @tab
string \$arr[col]["name"]	Le nom de la colonne col, avec col qui vaut entre 0 et \$arr["count"]-1. La valeur retournée peut être une chaîne vide (pour les colonnes dynamiquement générées). Les champs multiples SESAM sont linéarisés, et traités comme autant de colonnes, avec le même nom. @tab
string \$arr[col]["count"]	L'attribut "count" décrit le facteur de répétition quand la colonne a été déclarée comme un champs multiple. Généralement, cet attribut est à 1. La première colonne d'un champs multiple contient le nombre de répétitions, tandis que les colonnes suivantes ont un facteur de répétition mis à 1. Ceci peut être utilisé pour détecter les champs multiples. Reportez vous à l'exemple de la fonction sesam_query() pour avoir un exemple d'utilisation. @tab
string \$arr[col]["type"]	Type de variable PHP pour les données de la colonne col, où col vaut de 0 à \$arr["count"]-1. La valeur retournée peut être l'une de celles-ci :
"integer"	
"double"	
"string"	
, suivant le type de données SQL. Les champs multiples SESAM sont linéarisés et traités comme autant de colonnes ayant le même type PHP. @tab	
string \$arr[col]["sqltype"]	Type de données SQL de la colonne col, où col vaut de 0 à \$arr["count"]-1. La valeur retournée peut être l'une de celle-ci :
"CHARACTER"	
"VARCHAR"	
"NUMERIC"	
"DECIMAL"	
"INTEGER"	
"SMALLINT"	
"FLOAT"	
"REAL"	
"DOUBLE"	
"DATE"	
"TIME"	
"TIMESTAMP"	
, décrivant le type de données SQL. Les champs multiples SESAM sont linéarisés et traités comme autant de	

colonnes du même type. @tab	
string \$arr[col]["length"]	La taille de l'attribut, au sens SQL, de la colonne col, où col vaut de 0 à \$arr["count"]-1. La longueur est utilisée avec les champs "CHARACTER" et "VARCHAR", pour spécifier la taille maximale de la colonne. Les champs multiples SESAM sont linéarisés et traités comme autant de colonnes ayant la même taille SQL. @tab
string \$arr[col]["precision"]	La précision de la colonne col, au sens SQL, où col vaut de 0 à \$arr["count"]-1. La précision est utilisée avec les champs numériques et de date. Les champs multiples SESAM sont linéarisés et traités comme autant de colonnes ayant la même précision SQL. @tab
string \$arr[col]["scale"]	L'échelle de la colonne col, au sens SQL, où col vaut de 0 à \$arr["count"]-1. L'échelle est utilisée avec les champs numériques. Les champs multiples SESAM sont linéarisés et traités comme autant de colonnes ayant la même échelle SQL. @tab

Voir aussi [sesam_query\(\)](#), pour un exemple d'utilisation de [sesam_field_array\(\)](#).

10.64.15 [sesam_fetch_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sesam_fetch_row](#) (string *result_id*, int *whence* , int *offset*)

[sesam_fetch_row\(\)](#) retourne un tableau qui correspond à la ligne lue dans le résultat *result_id*, ou FALSE s'il n'y a plus de ligne.

Le nombre de colonnes du résultat est retourné dans un élément du tableau associatif retourné \$array["count"].

Comme certaines lignes peuvent être vides, la fonction [count\(\)](#) ne peut être utilisée avec le tableau ainsi retourné par [sesam_fetch_row\(\)](#).

result_id est un identifiant de résultat valide retourné par [sesam_query\(\)](#) (avec une requête de selection seulement!).

whence est un paramètre optionnel lors d'une opération de lecture sur un curseur à défilement, qui peut prendre une des valeurs suivantes :

Valeur	Constante	Signification
0	SESAM_SEEK_NEXT	Lecture séquentielle (après la lecture, la position est déplacé à SESAM_SEEK_NEXT) @tab
1	SESAM_SEEK_PRIOR	Lecture séquentielle à rebours (après la lecture, la position est déplacé à SESAM_SEEK_PRIOR) @tab
2	SESAM_SEEK_FIRST	Repositionnement au début (après la lecture, la position est déplacée à SESAM_SEEK_NEXT) @tab
3	SESAM_SEEK_LAST	Repositionnement à la fin (après la lecture, la position est déplacée à

		SESAM_SEEK_PRIOR) @tab
4	SESAM_SEEK_ABSOLUTE	Repositionnement absolu à <i>offset</i> (index commençant à 0. Après la lecture, la position est placée à SESAM_SEEK_ABSOLUTE, et le pointeur interne est auto-incrémenté). @tab
5	SESAM_SEEK_RELATIVE	Repositionnement relatif à <i>offset</i> , où <i>offset</i> peut être positif ou négatif @tab

Ce paramètre n'est valable que pour les curseurs à défilement.

Lors de l'utilisation de curseurs à défilement, le curseur peut être librement repositionné. Si le paramètre *whence* est omis, les valeurs par défaut seront utilisées (initialisées à : SESAM_SEEK_NEXT, et modifiée par [sesam_seek_row\(\)](#)). Si *whence* est fourni, sa valeur remplacera les valeurs par défaut.

offset est un paramètre optionnel qui n'est utilisé (et nécessaire) que si *whence* vaut soit SESAM_SEEK_RELATIVE ou SESAM_SEEK_ABSOLUTE. Ce paramètre n'est valable que pour les curseurs à défilement.

[sesam_fetch_row\(\)](#) lit une ligne de données dans le résultat *result_id*. La ligne est retournée sous forme d'un tableau (indexé de 0 à \$array["count"]-1). Les champs peuvent être vides : il faut vous assurer de leur existence en utilisant la fonction [isset\(\)](#). Le type de la valeur retournée dépend du type SQL déclaré dans la base (voir [10.64 SESAM](#) pour connaître les conversions utilisées). Les champs multiples SESAM sont linéarisés, et traités comme autant de colonnes.

Les prochains appels à [sesam_fetch_row\(\)](#) liront la prochaine ligne (ou la précédente, ou la n-ième, suivant le type de défilement) dans le résultat, ou FALSE, s'il n'y a plus de lignes.

Exemple avec [sesam_fetch_row\(\)](#)

```
<?php
$result = sesam_query ("SELECT * FROM phone\n".
                      " WHERE LASTNAME='".strtoupper($name)."' \n".
                      " ORDER BY FIRSTNAME", 1);

if (! $result) {
    ... erreur ...
}
// Affiche la table dans l'ordre inverse
print "<TABLE BORDER>\n";
$row = sesam_fetch_row ($result, SESAM_SEEK_LAST);
while (is_array($row)) {
    print " <TR>\n";
    for($col = 0; $col < $row["count"]; ++$col) {
        print " <TD>".htmlspecialchars($row[$col])."</TD>\n";
    }
    print " </TR>\n";
    // utilise la valeur implicite de SESAM_SEEK_PRIOR
    $row = sesam_fetch_row ($result);
}
print "</TABLE>\n";
sesam_free_result ($result);
?>
```

Voir aussi : [sesam_fetch_array\(\)](#) qui retourne un tableau associatif, et [sesam_fetch_result\(\)](#) qui retourne plusieurs lignes en même temps.

10.64.16 [sesam_fetch_array](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sesam_fetch_array](#) (string *result_id*, int *whence* , int *offset*)

[sesam_fetch_array\(\)](#) retourne un tableau qui correspond à la ligne lue dans le résultat *result_id*, ou FALSE si il n'y a pas d'autres lignes.

[sesam_fetch_array\(\)](#) est une version alternative de [sesam_fetch_row\(\)](#). Au lieu de stocker les données dans un tableau à indice numérique, il enregistre les données dans un tableau associatif, en utilisant les noms des champs comme clés.

result_id est un identifiant de résultat valide retourné par [sesam_query\(\)](#) (avec une requête de selection seulement!).

Pour connaître les valeurs valides des options *whence* et *offset*, reportez vous à [sesam_fetch_row\(\)](#).

[sesam_fetch_array\(\)](#) lit une ligne de données dans le résultat *result_id*. La ligne est retournée sous forme d'un tableau associatif. Chaque colonne est enregistrée avec leur nom comme index. Les noms des colonnes sont convertis en minuscules.

Les colonnes sans noms (par exemple, les résultats d'opérations arithmétiques) et les champs vides ne sont pas stockés dans ce tableau. De plus, si deux colonnes ont le même noms, la dernière colonne écrasera la précédente. Dans cette situation, utilisez de préférence [sesam_fetch_row\(\)](#) ou bien, faites un alias de la colonne.

```
SELECT TBL1.COL AS FOO, TBL2.COL AS BAR FROM TBL1, TBL2
```

Une gestion spéciale permet de lire les champs multiples, qui sinon, auraient toutes le même nom. Pour chaque colonne d'un champs multiple, le nom d'index est créé en ajoutant le numéro de sous-index à la suite du nom de la colonne. Ces sous indices sont numérotés à partir de 1.

```
CREATE TABLE ... ( ... MULTI(3) INT )
```

Les index associatifs utilisé pour les valeurs individuelles du champs multiple sont : "multi(1)", "multi(2)", et "multi(3)", respectivement.

Les prochains appels à [sesam_fetch_array\(\)](#) liront la prochaine ligne (ou la précédente, ou la n-ième, suivant les attributs de défilement), jusqu'à ce qu'il n'y ait plus de lignes.

Exemple avec [sesam_fetch_array\(\)](#)

```
<?php
$result = sesam_query ("SELECT * FROM phone\n".
    " WHERE LASTNAME='".strtoupper($name)."' \n".
    " ORDER BY FIRSTNAME", 1);

if (! $result) {
    ... error ...
}
// Affiche la table
print "<TABLE BORDER>\n";
while (($row = sesam_fetch_array ($result)) #38; count($row) 38;gt; 0) {
print " <TR>\n";
print " <TD>".htmlspecialchars($row["firstname"])."</TD>\n";
print " <TD>".htmlspecialchars($row["lastname"])."</TD>\n";
print " <TD>".htmlspecialchars($row["phoneno"])."</TD>\n";
print " </TR>\n";
}
```

```
print "</TABLE>\n";
sesam_free_result ($result);
?>
```

Voir aussi : [sesam_fetch_row\(\)](#) qui retourne un tableau numérique.

10.64.17 [sesam_seek_row](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [sesam_seek_row](#)(string *result_id*, int *whence*, int *offset*)

result_id est un indentifiant de résultat valide (requête de sélection, et curseur à défilement créé avec [sesam_query\(\)](#)).

whence modifie la valeur globale par défaut pour le type de défilement, spécifie le type de défilement à utiliser lors des opérations de lectures ultérieurs. Les valeurs valides sont les suivantes :

Valeur	Constante	Signification
0	SESAM_SEEK_NEXT	Lecture séquentielle (après la lecture, la position est déplacé à SESAM_SEEK_NEXT) @tab
1	SESAM_SEEK_PRIOR	Lecture séquentielle à rebours (après la lecture, la position est déplacé à SESAM_SEEK_PRIOR) @tab
2	SESAM_SEEK_FIRST	Repositionnement au début (après la lecture, la position est déplacée à SESAM_SEEK_NEXT) @tab
3	SESAM_SEEK_LAST	Repositionnement à la fin (après la lecture, la position est déplacée à SESAM_SEEK_PRIOR) @tab
4	SESAM_SEEK_ABSOLUTE	Repositionnement absolu à <i>offset</i> (index commençant à 0. Après la lecture, la position est placé à SESAM_SEEK_ABSOLUTE, et le pointeur interne est auto-incrémenté). @tab
5	SESAM_SEEK_RELATIVE	Repositionnement relatif à <i>offset</i> , où <i>offset</i> peut être positif ou négatif @tab

offset est optionnel. Il ne sert que lorsque *whence* vaut soit SESAM_SEEK_RELATIVE, soit SESAM_SEEK_ABSOLUTE.

10.64.18 [sesam_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sesam_free_result](#)(string *result_id*)

[`sesam_free_result\(\)`](#) libère les ressources réservées par la requête *result_id*. retourne FALSE en cas d'erreur.

10.65 Sessions

[\[Notes en ligne\]](#)

La gestion des sessions avec PHP est un moyen de sauver des informations entre deux accès. Cela permet notamment de construire des applications personnalisées, et d'accroître l'attrait de votre site.

Si vous connaissez déjà la gestion des sessions avec phplib, vous remarquerez que certains concepts sont similaires.

Chaque visiteur qui accède à votre site se voit assigner un numéro d'identifiant, appelé plus loin "identifiant de session". Celui ci est enregistré soit dans un cookie, chez le client, soit dans l'URL.

Les sessions vous permettront d'enregistrer des variables, pour les préserver et les réutiliser tout au long des requêtes. Lorsqu'un visiteur accède à votre site, PHP vérifiera automatiquement (si `session.auto_start` est à 1) ou manuellement (explicitement avec [`session_start\(\)`](#) ou implicitement avec [`session_register\(\)`](#)) si une session a déjà été ouverte. Si une telle session existe déjà, l'environnement précédent sera recréé.

Toutes les variables à enregistrer seront enregistrées sur le disque à la fin de chaque requête. Les variables enregistrées mais non définies seront marquées comme tel. Lors des accès ultérieurs, elles ne seront définies que si l'utilisateur le fait.

Les options `track_vars` et `gpc_globals` modifient la façon avec laquelle les variables sont rechargées.

Note : Depuis PHP 4.0.3, [7.1.1.27 ini.track-vars](#) est toujours activée.

Si `track_vars` est activée, et [7.1.1.23 ini.register-globals](#) est désactivée, alors les variables de session seront accessibles uniquement dans le tableau associatif global `$HTTP_STATE_VARS`. Les variables de sessions lues seront disponibles dans `$HTTP_STATE_VARS`.

Enregistrer une variable lorsque l'option [7.1.1.27 ini.track-vars](#) est activée

```
<?php
session_register("compte");
$HTTP_SESSION_VARS["compte"]++;
?>
```

Si [7.1.1.23 ini.register-globals](#) est activé, alors les variables de session seront placés dans les variables globales associées.

Enregistrer une variable lorsque [7.1.1.23 ini.register-globals](#) est activée

```
<?php
session_register("compte");
$compte++;
?>
```

Si les deux options [7.1.1.27 ini.track-vars](#) et [7.1.1.23 ini.register-globals](#) sont activées, alors les variables globales et `$HTTP_STATE_VARS` contiendront les valeurs de session.

Il y a deux modes de propagation de l'identifiant de session :

- Cookies
- Paramètre URL

Le module de session supporte les deux techniques. La méthode par cookies est optimale, mais étant donné le peu de fiabilité (les clients peuvent les refuser, ou les effacer), on ne peut pas se contenter de cette technique.

La deuxième méthode place l'identifiant de session directement dans l'URL.

PHP est capable de gérer ceci de manière transparente, lorsque vous le compilez avec l'option `--enable-trans-sid`. Dans ce cas, les URL relatives seront modifiées pour contenir l'identifiant de session automatiquement. Sinon, vous pouvez toujours utiliser la constante `SID`, qui sera définie si le client n'envoie pas le cookie approprié. `SID` prend la forme de `session_name=session_id`, ou bien, c'est une chaîne vide.

Note : *La fonction qui gérera l'écriture des données ne sera appelée qu'une fois le script aura envoyé toutes ses données. Ainsi, les affichages tentés par cette fonction ne pourront jamais être recus par le navigateur. Si un tel affichage est nécessaire, il est conseillé d'écrire les debugs dans un fichier.*

L'exemple suivant montre comment enregistrer une variable, et comment relier correctement des pages avec `SID`.

Compter le nombre de hit d'un utilisateur.

```
<?php
session_register("compteur");
$compteur++;
?>
Salut visiteur, vous avez vu cette page <?php echo $compteur; ?> times.<P>
<php?
# le <?=SID> est nécessaire pour transmettre l'identifiant de session
# au cas où les utilisateurs auraient inactivé les cookies
?>
```

Pour continuer, `<A HREF="nextpage.php?<?=SID"?>clique ici</?>`

Le `<?=SID>` n'est pas nécessaire, si l'option `--enable-trans-sid` a été utilisé pour compiler PHP.

Pour enregistrer ces informations dans une base de données, il vous faut utiliser la fonction [session_set_save_handler\(\)](#). Il faudra alors implémenter la fonction suivante pour l'adapter à MySQL ou toute autre base de données :

Le système de gestion des sessions dispose d'un grand nombre d'options, qui sont placées dans le fichier ``php.ini'`. En voici un survol rapide :

- `session.save_handler` définit les noms des fonctions qui seront utilisées pour enregistrer et retrouver les données associées à une session. Par défaut, les sessions sont enregistrées dans des fichiers.
- `session.save_path` définit l'argument qui est passé à la fonction de sauvegarde. Si vous utilisez la sauvegarde par fichier, cet argument est le chemin jusqu'au dossier où les fichiers sont créés. Par défaut, le dossier est `/tmp`.
Si le dossier que vous utilisez a les droits de lecture universels, comme ``/tmp'` (valeur par défaut), les autres utilisateurs du serveur peuvent aussi lire ces fichiers, et s'immiscer dans vos sessions.
- `session.name` spécifie le nom de la session, qui sera utilisé comme nom de cookie. Par défaut : `PHPSESSID`.
- `session.auto_start` indique qu'une session doit commencer automatiquement lors de la première requête. Par défaut à 0 (inactivé).
- `session.lifetime` fixe la durée de vie, en secondes, du cookie envoyé au client. La valeur 0 signifie "jusqu'à ce que le client soit fermé". Par défaut à 0 (inactivé).
- `session.serialize_handler` définit le nom de la fonction qui sera utilisée pour enregistrer et relire les données. Actuellement, c'est un format interne de PHP (nom : `php`) et `WDDX` (nom : `wddx`). `WDDX` n'est utilisable que si PHP a été compilé avec le [10.75 WDDX](#). Par défaut, c'est le mode `php` qui est sélectionné.
- `session.gc_probability` précise la probabilité que la routine `gc` (garbage collection) soit lancée, en pourcentage. Par défaut, 1.
- `session.gc_maxlifetime` fixe la durée, en secondes, au-delà de laquelle les données considérées comme inutiles seront supprimées.
- `session.referer_check` détermine si l'identifiant de session `ids` utilisé par des sites externes seront éliminés. Si les identifiants de sessions sont propagés avec la méthode des URL, des utilisateurs qui

n'en connaîtrait pas l'utilité risque de divulguer ces valeurs, et cela mènera à des problèmes de sécurité. Cette option y remédie. Par défaut : 0.

- `session.entropy_file` est le chemin jusqu'à une source externe (fichier) d'entropie, qui sera utilisée lors de la création de l'identifiant de session. Par exemple, `/dev/random` ou `/dev/urandom` qui sont disponibles sur de nombreux systèmes UNIX.
- `session.entropy_length` précise le nombre d'octets qui seront lus dans le fichier ci-dessus. Par défaut, 0 (inactif).
- `session.use_cookies` indique si le module doit utiliser des cookies pour enregistrer l'identifiant de session chez le client. Par défaut, 1 (activé).
- `session.cookie_path` spécifie le chemin à utiliser avec `session_cookie`. Par défaut, `/`.
- `session.cookie_domain` spécifie le domaine à utiliser avec `session_cookie`. Par défaut, rien du tout.
- `session.cache_limiter` spécifie le contrôle du cache, à utiliser avec les pages de sessions (`nocache/private/public`). Par défaut, `nocache`.
- `session.cache_expire` spécifie la durée de vie des pages de sessions cachées, en minutes, mais sans que cela n'ait d'effet sur le limiteur "nocache". Par défaut, 180.

Note : *La gestion des sessions a été ajoutée dans PHP 4.0.*

10.65.1 session_start

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool `session_start`

[PHP 4]

`session_start()` crée une session (ou continue la session courante, en fonction de l'identifiant de session passé par une variable GET ou par un cookie)

`session_start()` retourne toujours TRUE.

Note : `session_start()` a été ajoutée en PHP 4.0.

10.65.2 session_destroy

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool `session_destroy`

[PHP 4]

`session_destroy()` détruit toutes les données associées à la session courante.

`session_destroy()` retourne TRUE en cas de succès, et FALSE sinon.

10.65.3 session_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string `session_name` (string *name*)

[PHP 4]

`session_name()` retourne le nom de la session courante. Si *name* est fourni, le nom de la session changera, et prendra la valeur fournie.

Le nom de session fait référence à l'identifiant de session dans les cookies. Il ne doit contenir que des caractères alphanumériques; il doit être court et descriptif. (i.e. surtout pour les utilisateurs d'alertes de

cookie). Le nom de session est remis à une valeur par défaut, enregistrées dans `session.name` au moment du démarrage. Ainsi, vous devez appeler [session_name\(\)](#) à chaque requête (et avant [session_start\(\)](#) ou [session_register\(\)](#)).

Exemple avec session_name()

```
<?php
# Change le nom de la session à WebsiteID
$previous_name = session_name ("WebsiteID");
echo "L'ancien nom de la session était $previous_name<P>";
?>
```

Note : [session_name\(\)](#) a été ajoutée en PHP 4.0.

10.65.4 session_module_name

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_module_name](#)(string *module*)
[PHP 4]

[session_module_name\(\)](#) affecte et/ou retourne le module courant de session courante. Si *module* est fourni, ce module sera utilisé à la place du courant. Note : [session_module_name\(\)](#) a été ajoutée en PHP 4.0.

10.65.5 session_save_path

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_save_path](#)(string *path*)
[PHP 4]

[session_save_path\(\)](#) retourne le chemin du dossier utilisé pour enregistrer les données de sessions. Si *path* est fourni, le chemin prendra alors la valeur fournie. Note : *Sur certains systèmes d'exploitation, il vous faudra peut être fournir un chemin vers un système de sauvegarde qui peut gérer de grandes quantités de petits fichiers efficacement : par exemple, sous Linux, reiserfs peut être plus efficace que ext2fs.*

Note : [session_save_path\(\)](#) a été ajoutée en PHP 4.0.

10.65.6 session_id

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_id](#)(string *id*)
[PHP 4]

[session_id\(\)](#) retourne l'identifiant de session courante. Si *id* est fourni, il remplacera l'identifiant courant de la session.

La constante `SID` peut aussi être utilisée pour retrouver le nom de la session courante et son identifiant, comme chaîne à ajouter dans les URL. Note : [session_id\(\)](#) a été ajoutée en PHP 4.0.

[10.65.7 session_register](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_register](#)(mixed *name*, mixed ...)

[PHP 4]

[session_register\(\)](#) enregistre une variable avec le nom *name* dans la session courante.

[session_register\(\)](#) accepte un nombre d'arguments variable, qui peuvent être des chaînes représentant le nom de la variable, ou bien un tableau, représentant des chaînes ou d'autre tableau (récursif).

[session_register\(\)](#) retourne TRUE lorsque la variable est correctement enregistrée. Note : [session_register\(\)](#) a été ajoutée en PHP 4.0.

[10.65.8 session_unregister](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_unregister](#)(string *name*)

[PHP 4]

[session_unregister\(\)](#) supprime la variable nommée *name* dans la session courante.

[session_unregister\(\)](#) retourne TRUE lorsque la variable a été correctement supprimée de la session. Note : [session_unregister\(\)](#) a été ajoutée en PHP 4.0.

[10.65.9 session_unset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [session_unset](#)

[PHP 4 >= 4.0b4]

[session_unset\(\)](#) détruit toutes les variables de session couramment enregistrées.

[10.65.10 session_is_registered](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_is_registered](#)(string *name*)

[PHP 4]

[session_is_registered\(\)](#) retourne TRUE s'il y a une variable du nom de *name* enregistrée dans la session courante. Note : [session_is_registered\(\)](#) a été ajoutée en PHP 4.0.

[10.65.11 session_get_cookie_params](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [session_get_cookie_params](#)

[PHP 4 >= 4.0RC2]

[session_get_cookie_params\(\)](#) retourne un tableau avec les paramètres du cookie de la session courante. Le tableau contient les éléments suivants :

- "lifetime" – La durée de vie du cookie.
- "path" – Le chemin de stockage du cookie.
- "domain" – Le domaine du cookie.

10.65.12 session_set_cookie_params

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [session_set_cookie_params](#)(int *lifetime* , string *path* , string *domain*)
[PHP 4 >= 4.0b4]

[session_set_cookie_params\(\)](#) modifie les paramètres du cookie de session, tel qu'ils ont été définis dans le fichier ``php.ini'`. L'effet de cette fonction ne dure que le temps du script.

10.65.13 session_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_decode](#)(string *data*)
[PHP 4]

[session_decode\(\)](#) décode les données de session à partir de la chaîne *data*, et affecte les valeurs des variables de session. Note : [session_decode\(\)](#) a été ajoutée en PHP 4.0.

10.65.14 session_encode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [session_encode](#)
[PHP 4]

[session_encode\(\)](#) retourne les données de session dans une chaîne. Note : [session_encode\(\)](#) a été ajoutée en PHP 4.0.

10.65.15 session_set_save_handler

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [session_set_save_handler](#)(string *open*, string *close*, string *read*, string *write*, string *destroy*, string *gc*)
[PHP 4 >= 4.0b4]

[session_set_save_handler\(\)](#) définit les fonctions utilisateurs de stockage et chargement des sessions. Cela est particulièrement pratique pour spécifier une autre méthode de stockage que celle fournie en standard avec

PHP. Notamment, il est possible de stocker les sessions dans une base de données.

Note : Vous devez donner à l'option de configuration ***session.save_handler*** la valeur de ***user*** dans votre fichier `php.ini` pour que [*session_set_save_handler\(\)*](#) soit effective.

L'exemple suivant fournit un exemple de stockage de session dans un fichier, similaire aux fonctions standard de PHP. Cet exemple peut être facilement étendu pour utiliser un stockage en base de données, en utilisant votre base préférée.

Exemple avec session_set_save_handler()

```
<?php
function open ($save_path, $session_name) {
global $sess_save_path, $sess_session_name;
    $sess_save_path = $save_path;
    $sess_session_name = $session_name;
return(TRUE);
}
function close() {
return(TRUE);
}
function read ($id) {
global $sess_save_path, $sess_session_name;
    $sess_file = "$sess_save_path/sess_$id";
    if ($fp = @fopen($sess_file, "r")) {
        $sess_data = fread($fp, filesize($sess_file));
return($sess_data);
    } else {
return("");
    }
}
function write ($id, $sess_data) {
global $sess_save_path, $sess_session_name;
    $sess_file = "$sess_save_path/sess_$id";
    if ($fp = @fopen($sess_file, "w")) {
return(fwrite($fp, $sess_data));
    } else {
return(FALSE);
    }
}
function destroy ($id) {
global $sess_save_path, $sess_session_name;
    $sess_file = "$sess_save_path/sess_$id";
return(@unlink($sess_file));
}
/*****
 * ATTENTION - Vous devez implémenter une routine
 * d'entretien des sessions ici.
 *****/
function gc ($maxlifetime) {
return TRUE;
}
session_set_save_handler ("open", "close", "read", "write", "destroy", "gc");
session_start();
// utilisez vos sessions normalement
?>
```

10.65.16 session_cache_limiter

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [session_cache_limiter](#)(string *cache_limiter*)

[PHP 4 >= 4.0.3]

[session_cache_limiter\(\)](#) retourne le nom du limiteur de cache courant. Si *cache_limiter* est spécifié, le nom du limiteur de cache est remplacé par cette nouvelle valeur.

Le limiteur de cache contrôle l'envoi des entête HTTP envoyés au client. Ces entêtes déterminent les règles de mise en cache des pages. En utilisant la valeur de nocache, par exemple, vous désactiverez la mise en cache coté client. La valeur de public, cependant, le permettra. private aussi, tout en étant légèrement plus restrictive que public.

Le limiteur de cache est remis à sa valeur par défaut, stockée dans session.cache_limiter, initialisée au lancement. Vous devrez donc appeler [session_cache_limiter\(\)](#) pour chaque requête (et avant l'appel à [session_start\(\)](#)).

Exemples avec session_cache_limiter()

```
<?php
# Met le limiteur de cache à 'private'
session_cache_limiter('private');
$cache_limiter = session_cache_limiter();
echo "Le limiteur de cache vaut actuellement $cache_limiter<P>";
?>
```

Note : [session_cache_limiter\(\)](#) a été ajoutée dans PHP 4.0.3.

10.66 Mémoire partagée

[\[Notes en ligne\]](#)

Shmop est un ensemble de fonctions simples pour gérer la mémoire partagée avec PHP (lecture, écriture, création et suppressions de segments de mémoire partagée UNIX). Ces fonctions ne fonctionnent pas sous Windows, car ce système d'exploitation ne supporte pas la mémoire partagée. Pour utiliser les fonctions shmop, compilez PHP avec l'option `--enable-shmop` parameter.

Introduction à la mémoire partagée

```
<?php
// Crée 100 octets de mémoire partagée avec
// un identifiant système "0xff3"
$shm_id = shm_open(0xff3, "c", 0644, 100);
if(!$shm_id) {
echo "Impossible de créer la mémoire partagée\n";
}
// Lire la taille de la mémoire partagée
$shm_size = shm_size($shm_id);
echo "Un bloc de SHM de taille ".$shm_size. " a été créé.\n";
// Ecriture d'une chaîne de test dans ce segment
$shm_bytes_written = shm_write($shm_id, "mon bloc de mémoire partagée", 0);
if($shm_bytes_written != strlen("mon bloc de mémoire partagée")) {
echo "Impossible d'écrire toutes les données en mémoire\n";
}
// Lecture du segment
```

```

$my_string = shm_read($shm_id, 0, $shm_size);
if(!$my_string) {
echo "Impossible de lire toutes les données en mémoire\n";
}
echo "Les données mis en mémoire partagées sont : ".$my_string."\n";
//Maintenant, effaçons le bloc, et fermons le segment de mémoire
if(!shm_delete($shm_id)) {
echo "Impossible d'effacer le segment de mémoire";
}
shm_close($shm_id);
?>

```

10.66.1 shm_open

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_open](#)(int *key*, string *flags*, int *mode*, int *size*)

[PHP 4 >= 4.0.3]

[shm_open\(\)](#) peut créer ou ouvrir un bloc de mémoire partagée.

[shm_open\(\)](#) prend 4 paramètres: la clé, qui sera l'identifiant système pour le bloc. Ce paramètre peut être passé comme un décimal ou un hexadécimal. Le deuxième paramètre est un groupe d'options :

- "a" pour accès (utilise IPC_EXCL) utilisez cette option pour ouvrir un bloc déjà existant.
- "c" pour création (utilise IPC_CREATE) utilisez cette option pour créer un nouveau bloc.

Le troisième paramètre est le mode, c'est à dire les permissions que vous donnez à ce bloc. Ce sont les mêmes que pour les fichiers. Ces permissions doivent être passées sous forme d'octal (i.e. 0644). Le dernier paramètre est la taille du bloc de mémoire, en octets. Note : *Note: Les troisième et quatrième paramètres doivent être passés à 0 si vous voulez ouvrir un bloc de mémoire partagée déjà existant. En cas de succès*

[shm_open\(\)](#) retourne un identifiant que vous pouvez utiliser pour accéder à la mémoire que vous venez de créer.

Créer un nouveau bloc

```

<?php
$shm_id = shm_open(0x0fff, "c", 0644, 100);
?>

```

Cet exemple ouvre un nouveau bloc de mémoire partagée, dont l'identifiant est 0x0fff.

10.66.2 shm_read

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [shm_read](#)(int *shm_id*, int *start*, int *count*)

[PHP 4 >= 4.0.3]

[shm_read\(\)](#) lit une chaîne dans un bloc de mémoire partagée.

[shm_read\(\)](#) prend 3 paramètres: *shm_id*, qui est un identifiant de mémoire partagée, créé par [shm_open\(\)](#).

start qui est la position à partir de laquelle on commence à lire dans la mémoire et **count**, le nombre d'octets à lire.

Lire un bloc de mémoire partagée

```
<?php
$shm_data = shm_read($shm_id, 0, 50);
?>
```

Cet exemple lit 50 octets dans le bloc de mémoire partagée \$shm_data.

10.66.3 shm_write

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_write](#)(int *shm_id*, string *data*, int *offset*)

[PHP 4 >= 4.0.3]

[shm_write\(\)](#) écrit une chaîne dans un bloc de mémoire partagée.

[shm_write\(\)](#) prend 3 paramètres: *shm_id*, qui est un identifiant de mémoire partagée, créé par [shm_open\(\)](#), *data* qui est la chaîne à écrire dans la mémoire et *offset*, la position à partir de laquelle il faut commencer à écrire.

Ecrire un bloc de mémoire partagée

```
<?php
$shm_bytes_written = shm_write($shm_id, $my_string, 0);
?>
```

Cet exemple écrit les données de la chaîne \$my_string dans un bloc de mémoire partagée.

\$shm_bytes_written représentera le nombre d'octets écrits.

10.66.4 shm_size

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_size](#)(int *shm_id*)

[PHP 4 >= 4.0.3]

[shm_size\(\)](#) sert à connaître la taille, en octets, d'un bloc de mémoire partagée.

[shm_size\(\)](#) prend comme argument *shm_id*, un identifiant de bloc de mémoire partagée créé par [shm_open\(\)](#), et retourne un entier, qui représente la taille de ce bloc.

Lire la taille d'un bloc de mémoire partagée

```
<?php
$shm_size = shm_size($shm_id);
?>
```

Cet exemple lit la taille du bloc identifié par \$shm_id, et le place dans \$shm_size.

10.66.5 shm_delete

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_delete](#)(int *shm_id*)

[PHP 4 >= 4.0.3]

[shm_delete\(\)](#) sert à détruire un bloc de mémoire partagée.

[shm_delete\(\)](#) prend un identifiant de mémoire partagée *shm_id*, créé par [shm_open\(\)](#). En cas de succès, la fonction retourne 1, et sinon, 0.

Effacement d'un bloc de mémoire partagée

```
<?php
shm_delete($shm_id);
?>
```

Ce exemple efface le bloc de mémoire partagée identifié par \$shm_id.

10.66.6 shm_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [shm_close](#)(int *shm_id*)

[PHP 4 >= 4.0.3]

[shm_close\(\)](#) sert à fermer un bloc de mémoire partagée.

[shm_close\(\)](#) prend un identifiant de mémoire partagée, *shm_id*, créé par [shm_open\(\)](#).

Fermeture d'un bloc de mémoire partagée

```
<?php
shm_close($shm_id);
?>
```

Cet exemple ferme le bloc de mémoire partagée identifié par \$shm_id.

10.67 SNMP

[\[Notes en ligne\]](#)

Afin de pouvoir utiliser les fonctions **SNMP** sous Unix, vous aurez besoin d'installer le package [UCD SNMP](#). Sous Windows ces fonctions ne sont disponibles que sous NT, et pas sous Win95/98.

Important : Afin d'utiliser le package UCD **SNMP**, vous devez mettre la variable NO_ZEROLENGTH_COMMUNITY à 1 avant de compiler. Après avoir configuré UCD **SNMP**, éditez le fichier config.h et recherchez la valeur NO_ZEROLENGTH_COMMUNITY. Décommentez la ligne avec le #define. Cela doit ressembler à ceci :

```
#define NO_ZEROLENGTH_COMMUNITY 1
```

Si vous avez des erreurs "segmentation faults", lors de l'utilisation des commandes *SNMP*, c'est que vous n'avez pas suivi les recommandations précédentes. Si vous ne voulez pas recompiler UCD *SNMP*, vous pouvez aussi recompiler PHP avec l'option `--enable-ucd-snmp-hack` qui évitera cette erreur.

[10.67.1 snmpget](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [snmpget](#)(string *hostname*, string *community*, string *object_id*, int *timeout*, int *retries*)

[PHP 3, PHP 4]

[snmpget\(\)](#) retourne un objet *SNMP* en cas de succès, et FALSE en cas d'erreur.

[snmpget\(\)](#) sert à lire une valeur d'un objet *SNMP* représenté par *object_id*. L'agent *SNMP* est défini par *hostname* et la communauté de lecture est spécifiée par le paramètre *community*.

```
$syscontact = snmpget("127.0.0.1", "public", "system.SysContact.0")
```

[10.67.2 snmpset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [snmpset](#)(string *hostname*, string *community*, string *object_id*, string *type*, mixed *value*, int *timeout*, int *retries*)

[PHP 3 >= 3.0.12, PHP 4 >= 4.0b2]

[snmpset\(\)](#) modifie la valeur de l'objet *SNMP* spécifié, en retournant TRUE en cas de succès et FALSE en cas d'erreur.

[snmpset\(\)](#) sert à affecter une valeur donnée à un objet *SNMP*, référencé par *object_id*. L'agent *SNMP* est défini par *hostname* et la communauté de lecture est spécifiée par le paramètre *community*.

[10.67.3 snmpwalk](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [snmpwalk](#)(string *hostname*, string *community*, string *object_id*, int *timeout*, int *retries*)

[PHP 3, PHP 4]

[snmpwalk\(\)](#) retourne un tableau d'objets *SNMP*, en commençant à partir de *object_id* comme racine, ou FALSE en cas d'erreur.

[snmpwalk\(\)](#) sert à lire toutes les valeurs d'un agent *SNMP*, défini par *hostname*. *community* définit la communauté de lecture de l'agent. Un objet (*object_id* = null) sert de racine à l'arbre d'objet *SNMP* et tous les objets sous cette racine sont retournés dans un tableau. Si *object_id* est spécifié, tous les objets *SNMP* sous cet objet sont retournés.

```
<?php
$a = snmpwalk("127.0.0.1", "public", "");
```



```
?>
```

La fonction ci-dessus va retourner tous les objets *SNMP* d'un agent *SNMP* qui fonctionnerait sur l'hôte local (localhost). Il suffit alors de faire une boucle pour travailler avec chacun des objets.

```
<?php
for ($i=0; $i<count($a); $i++) {
echo $a[$i];
}
?>
```

10.67.4 snmpwalkoid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [snmpwalkoid](#) (string *hostname*, string *community*, string *object_id*, int *timeout* , int *retries*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[snmpwalkoid\(\)](#) retourne un tableau associatif, avec les identifiants d'objet et les objets associés, pour tous les objets situés sous la racine *object_id*, ou FALSE en cas d'erreur.

[snmpwalkoid\(\)](#) sert à lire tous les identifiants d'objet, et leur valeurs respectives, depuis un serveur *SNMP*. *community* indique la communauté de lecture pour cet agent. Un *object_id* null signifie qu'il faut utiliser la racine de l'arbre *SNMP* et tous les objets sous cet arbre seront retournés. Si *object_id* est spécifié, tous les objets *SNMP* situés sous cet objet seront retournés.

La fonction ci-dessous va lire tous les objets de l'agent *SNMP* qui fonctionne sur l'hôte local. Il est alors possible de les passer en revue avec une boucle : l'existence de [snmpwalkoid\(\)](#) et [snmpwalk\(\)](#) est une question d'évolution. Ces deux fonctions sont fournies pour des raisons de compatibilité ascendante.

```
<?php
$a = snmpwalkoid("127.0.0.1", "public", "");
?>
```

La fonction ci-dessous va lire tous les objets de l'agent *SNMP* qui fonctionne sur l'hôte local. Il est alors possible de les passer en revue avec une boucle :

```
for (reset($a); $i = key($a); next($a)) {
echo "$i: $a[$i]<br>\n";
}
```

10.67.5 snmp_get_quick_print

[\[Notes en ligne\]](#) [\[Exemples\]](#)

boolean [snmp_get_quick_print](#) (void)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[snmp_get_quick_print\(\)](#) retourne la valeur courante, stockée dans la librairie UCD, de l'option `quick_print`. Par défaut, `quick_print` est inactivée.

```
<?php
$quickprint = snmp_get_quick_print();
?>
```

L'exemple ci-dessus devrait retourner `FALSE`, si `quick_print` est inactivée et, `TRUE` si `quick_print` est activée.

[snmp_get_quick_print\(\)](#) est seulement disponible avec la librairie UCD *SNMP*. Cette fonction n'est pas disponible avec la librairie Windows *SNMP*.

Voir : [snmp_set_quick_print\(\)](#) pour une description complète de l'affichage de `quick_print`.

10.67.6 [snmp_set_quick_print](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [snmp_set_quick_print](#)(boolean *quick_print*)
[PHP 3 >= 3.0.8, PHP 4 >= 4.0b2]

[snmp_set_quick_print\(\)](#) fixe la valeur de l'option `quick_print` de la librairie UCD *SNMP*. Lorsqu'elle a la valeur de (1), la librairie *SNMP* retournera des valeurs 'rapides'. Cela signifie que seule, la valeur sera retournée. Lorsqu'elle a la valeur de (0), la librairie va afficher d'autres informations (telles que l'adresse IP (IpAddress) ou OID). De plus, si `quick_print` n'est pas activée, la librairie affichera aussi des valeurs hexadécimales supplémentaires pour toutes les chaînes de trois caractères, ou moins. Modifier `quick_print` est plus fréquent lorsqu'on utilise les valeurs retournées que lorsqu'on les affiche.

```
snmp_set_quick_print(0);
$a = snmpget("127.0.0.1", "public", ".1.3.6.1.2.1.2.2.1.9.1");
echo "$a<BR>\n";
snmp_set_quick_print(1);
$a = snmpget("127.0.0.1", "public", ".1.3.6.1.2.1.2.2.1.9.1");
echo "$a<BR>\n";
```

La première valeur affichée sera : 'Timeticks: (0) 0:00:00.00', tandis qu'avec `quick_print` activée, seul '0:00:00.00' sera affiché.

Par défaut, UCD *SNMP* retourne des valeurs détaillées, et `quick_print` sert à ne retourner que la valeur. Actuellement, les chaînes sont toujours retournées avec des guillemets supplémentaires. Ceci sera corrigé ultérieurement.

[snmp_set_quick_print\(\)](#) ne fonctionne qu'avec la librairie UCD *SNMP*. [snmp_set_quick_print\(\)](#) n'est pas disponible avec la librairie Windows *SNMP*.

10.68 [Sockets](#)

[\[Notes en ligne\]](#)

L'extension `socket` implémente une interface bas niveau avec les fonctions de communication par socket. Cela permet de mettre en place un serveur aussi bien qu'un client.

Les fonctions `socket` décrites ici sont rassemblées dans une extension PHP. Pour être activées, il faut utiliser l'option de compilation `--enable-sockets` au script `configure`.

Pour une interface client plus générique, reportez vous à [fsockopen\(\)](#) et [pfsockopen\(\)](#).

Lorsque vous utiliserez les fonctions de sockets qui sont décrites ici, gardez bien à l'esprit que même si elles ont souvent des noms identiques aux fonctions C, elles ont souvent des prototypes différents. Lisez attentivement la documentation pour éviter les confusions.

Cela dit, ceux qui n'ont pas l'habitude de la programmation avec les sockets pourront trouver beaucoup de documentation pertinente dans les pages de manuel Unix, et de nombreux tutoriel de programmation C sur le web, dont la plus part peuvent être repris après de légères modifications, en PHP.

Exemple de programmation Socket : serveur TCP/IP

Cet exemple est un serveur perroquete : tout ce que vous lui envoyez vous est retourné. Changez les variables **address** et **port** pour les adapter à votre configuration, et lancez le script. Vous pouvez vous connecter au serveur avec une commande telle que telnet 192.168.1.53 10000 (avec l'adresse et le port qui sont ceux de votre configuration). Pour vous déconnecter, tapez 'quit'.

```
<?php
error_reporting(E_ALL);
/* On autorise le script à attendre les connexions indéfiniment. */
set_time_limit(0);
/* Modifiez ces valeurs pour qu'elles soient celles de votre configuration */
$address = '192.168.1.53';
$port = 10000;
if (($sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
echo "socket() a échoué : raison : " . strerror($sock) . "\n";
}
if (($ret = bind($sock, $address, $port)) < 0) {
echo "bind() a échoué : raison: " . strerror($ret) . "\n";
}
if (($ret = listen($sock, 5)) < 0) {
echo "listen() a échoué : raison: " . strerror($ret) . "\n";
}
do {
if (($msgsock = accept_connect($sock)) < 0) {
echo "accept_connect() a échoué : raison : " . strerror($msgsock) . "\n";
break;
}
do {
$buf = '';
$ret = read($msgsock, $buf, 2048);
if ($ret < 0) {
echo "read() a échoué : raison : " . strerror($ret) . "\n";
break 2;
}
if ($ret == 0) {
break 2;
}
$buf = trim($buf);
if ($buf == 'quit') {
close($msgsock);
break 2;
}
$talkback = "PHP: Vous avez dit '$buf'.\n";
write($msgsock, $talkback, strlen($talkback));
echo "$buf\n";
} while (TRUE);
}
```

```
close($msgsock);
} while (TRUE);
close($sock);
?>
```

Exemple avec les sockets : Client TCP/IP

Cet exemple est un client HTTP basique. Il se connecte à une page envoie les entêtes (requête HEAD), affiche le retour, et quitte.

```
<?php
error_reporting(E_ALL);
echo "<h2>TCP/IP Connection</h2>\n";
/* Demande le port du service WWW. */
$service_port = getservbyname('www', 'tcp');
/* Demande l'IP du serveur de destination. */
$address = gethostbyname('www.php.net');
/* Crée la connexion TCP/IP. */
$socket = socket(AF_INET, SOCK_STREAM, 0);
if ($socket < 0) {
echo "socket() a échoué : raison : " . strerror($socket) . "\n";
} else {
    "socket() réussi: " . strerror($socket) . "\n";
}
echo "Connexion à '$address' on port '$service_port'...";
$result = connect($socket, $address, $service_port);
if ($result < 0) {
echo "connect() a échoué : raison : : ($result) " . strerror($result) . "\n";
} else {
echo "OK.\n";
}
$in = "HEAD / HTTP/1.0\r\n\r\n";
$out = '';
echo "Envoi des entêtes HTTP HEAD...";
write($socket, $in, strlen($in));
echo "OK.\n";
echo "Lecture de la réponse :\n\n";
while (read($socket, $out, 2048)) {
echo $out;
}
echo "Fermeture de la socket...";
close($socket);
echo "OK.\n\n";
?>
```

10.68.1 accept connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [accept connect](#) (int *socket*)

[PHP 4 >= 4.0.2]

Une fois que la socket **socket** a été créé, avec la fonction [socket\(\)](#), liée à un nom avec [bind\(\)](#), et mise en attente de connexion avec [listen\(\)](#), [accept_connect\(\)](#) accepte les connexions sur la socket **socket**. Une fois que la connexion est faite, un nouveau pointeur de socket est retourné, pour utilisation ultérieure. Si il n'y a pas de connexion en attente, [accept_connect\(\)](#) se bloquera jusqu'à une connexion soit disponible. Si **socket** a été configurée comme non-bloquante, avec [socket_set_blocking\(\)](#), une erreur sera retournée.

Le pointeur de socket retourné par [accept_connect\(\)](#) ne peut plus accepter de nouvelles connexion. La socket originale, **socket**, reste ouverte, et peut être réutilisée.

Retourne une nouveau pointeur de socket en cas de succès, ou une erreur négative en cas d'erreur. Ce code peut être passé à la fonction [strerror\(\)](#) pour obtenir un message d'erreur lisible.

Voir aussi [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), [socket\(\)](#), et [strerror\(\)](#).

10.68.2 bind

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [bind](#) (int *socket*, string *address*, int *protocol*)

[PHP 4 >= 4.0.2]

[bind\(\)](#) lie le nom *address*, à la socket **socket**, qui doit être une socket valide, créée avec [socket\(\)](#).

address peut être une adresse IP numérique (e.g. 127.0.0.1), si la socket est de la famille AF_INET; ou bien un chemin d'un domaine UNIX, si la socket est de la famille des AF_UNIX.

port sert uniquement dans le cas des sockets de type AF_INET et désigne le port de connexion sur l'hôte distant.

Retourne zéro en cas de succès, et une erreur négative en cas d'échec. Ce code peut être passé à la fonction [strerror\(\)](#) pour obtenir un message d'erreur lisible.

Voir aussi [accept_connect\(\)](#), [connect\(\)](#), [listen\(\)](#), [socket\(\)](#), et [strerror\(\)](#).

10.68.3 close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [close](#) (int *socket*)

[PHP 4]

[close\(\)](#) ferme le fichier (ou la socket) **socket**.

Notez que [close\(\)](#) ne doit pas être utilisée avec des pointeurs de fichiers créé par [fopen\(\)](#), [popen\(\)](#),

[fsockopen\(\)](#), ou [pfsockopen\(\)](#); elle ne sert que pour les sockets créées avec [socket\(\)](#) ou [accept_connect\(\)](#).

Retourne TRUE en cas de succès, ou FALSE en cas d'erreur (i.e., **socket** est invalide).

Voir aussi [bind\(\)](#), [listen\(\)](#), [socket\(\)](#), et [strerror\(\)](#).

10.68.4 connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [connect](#) (int *socket*, string *address*, int *port*)

[PHP 4 >= 4.0.2]

Initie une connexion avec la socket **socket**, qui doit être une socket valide, créée avec [socket\(\)](#).

address peut être une adresse IP numérique (e.g. 127.0.0.1), si la socket est de la famille AF_INET; ou bien un chemin d'un domaine UNIX, si la socket est de la famille des AF_UNIX.

port sert uniquement dans le cas des sockets de type AF_INET et désigne le port de connexion sur l'hôte

distant.

Retourne zéro en cas de succès, et une erreur négative en cas d'échec. Ce code peut être passé à la fonction [strerror\(\)](#) pour obtenir un message d'erreur lisible.

Voir aussi [bind\(\)](#), [listen\(\)](#), [socket\(\)](#), et [strerror\(\)](#).

10.68.5 listen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [listen](#)(int *socket*, int *backlog*)

[PHP 4 >= 4.0.2]

Une fois que la socket *socket* a été créée avec [socket\(\)](#) et liée avec [bind\(\)](#), elle peut être mise en attente de connexion entrante. Un maximum de *backlog* connexion entrantes seront mises en attente de traitement.

[listen\(\)](#) ne fonctionne qu'avec des sockets de type SOCK_STREAM et SOCK_SEQPACKET.

Retourne zéro en cas de succès, et une erreur négative en cas d'échec. Ce code peut être passé à la fonction [strerror\(\)](#) pour obtenir un message d'erreur lisible.

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [socket\(\)](#), et [strerror\(\)](#).

10.68.6 read

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [read](#)(int *socket_des*, string *&buffer*, int *length*)

[PHP 4]

[read\(\)](#) lit sur la socket *socket_des* créée avec [accept_connect\(\)](#), et place le résultat dans le buffer *&buffer*, *length* octets. Vous pouvez aussi utiliser \n, \t ou \0 pour terminer la lecture. Le nombre d'octets lus est retourné.

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), [strerror\(\)](#), [socket_get_status\(\)](#) et [write\(\)](#).

10.68.7 socket

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [socket](#)(int *domain*, int *type*, int *protocol*)

[PHP 4 >= 4.0.2]

Crée un point de communication, dit socket, et retourne un pointeur de socket.

domain représente le domaine. Actuellement, ce peut être AF_INET et AF_UNIX.

type sélectionne le type de socket. Il peut prendre les valeurs suivantes : SOCK_STREAM, SOCK_DGRAM, SOCK_SEQPACKET, SOCK_RAW, SOCK_RDM, ou SOCK_PACKET.

protocol choisit le protocole.

Retourne un pointeur de socket valide en cas de succès, et une erreur négative en cas d'échec. Ce code peut être passé à la fonction [strerror\(\)](#) pour obtenir un message d'erreur lisible.

For more information on the usage of [socket\(\)](#), as well as on the meanings of the various parameters, see the Unix man page ``socket (2)'`.

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), et [strerror\(\)](#).

10.68.8 [strerror](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strerror](#)(int *errno*)

[PHP 4 >= 4.0.2]

[strerror\(\)](#) prend comme paramètre *errno* la valeur négative de retour d'une fonction de socket, et retourne l'explication correspondante au format texte. Cela facilite grandement la recherche d'erreur. Par exemple, au lieu d'être bloqué par une erreur '-111', et de devoir en rechercher la signification dans les fichiers systèmes, il suffit de la passer à [strerror\(\)](#), pour savoir ce qui s'est passé.

Exemple avec strerror()

```
<?php
if (($socket = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
echo "socket() a échoué : raison: " . strerror($socket) . "\n";
}
if (($ret = bind($socket, '127.0.0.1', 80)) < 0) {
echo "bind() a échoué : raison: " . strerror($ret) . "\n";
}
?>
```

Le résultat de l'exemple ci dessus (en supposant que le script n'est pas exécuté avec les droits du root) :

```
bind() a échoué : raison : Permission denied
```

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#) et [socket\(\)](#).

10.68.9 [write](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [write](#)(int *socket_des*, string *&buffer*, int *length*)

[PHP 4 >= 4.0.2]

[write\(\)](#) écrit sur la socket *socket_des*, *length* octets issus du buffer *&buffer*.

Voir aussi [accept_connect\(\)](#), [bind\(\)](#), [connect\(\)](#), [listen\(\)](#), [read\(\)](#), [strerror\(\)](#) et [socket_get_status\(\)](#).

10.69 Chaîne de caractères

[\[Notes en ligne\]](#)

Ces fonctions permettent la manipulations de chaînes de caractères. Certaines sections plus spécialisées sont disponibles dès les sections sur les expressions régulières et dans la section URL.

Pour plus de détails sur le comportement des chaînes de caractères, notamment concernant les guillemets simples ou doubles, et les séquences d'échappement, reportez vous à [9.8.3 Les chaînes de caractères](#), dans le chapitre [9.8 Types](#).

10.69.1 addcslashes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [addcslashes](#) (string *str*, string *charlist*)

[PHP 4 >= 4.0b4]

Retourne une chaîne avec des antislash devant les caractères qui sont dans la liste *charlist*. Les caractères \n, \r etc. sont échappés. En langage C, les caractères avec un code ASCII inférieur à 32 ou supérieur à 126 sont convertis en représentation octale. Faites bien attention lorsque vous échappez des caractères alpha-numériques. Vous pouvez spécifier un intervalle dans *charlist* comme "\0..\37", qui échappera les caractères compris dans cet intervalle.

Exemple avec addcslashes()

```
<?php
$escaped = addcslashes($no_echappe, "\0..\37!@\177..\377");
?>
```

Note : [addcslashes\(\)](#) a été ajouté dans PHP 4.0b3-dev.

Voir aussi [stripslashes\(\)](#), [stripslashes\(\)](#), [htmlspecialchars\(\)](#) et [quotemeta\(\)](#).

10.69.2 addslashes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [addslashes](#) (string *str*)

[PHP 3, PHP 4]

Retourne une chaîne avec des antislash devant chaque caractère qui a en a besoin pour être inséré dans une requête de base de données. Ces caractères sont guillemets simples ('), guillemets doubles ("), antislash (\) et NULL (la valeur nulle).

Voir aussi [stripslashes\(\)](#), [htmlspecialchars\(\)](#) et [quotemeta\(\)](#).

10.69.3 bin2hex

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [bin2hex](#) (string *str*)

[PHP 3 >= 3.0.9, PHP 4]

Retourne une chaîne ASCII contenant la représentation hexadécimale de *str*. La conversion est faite avec le bit de poids fort en premier.

10.69.4 chop

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [chop](#) (string *str*)

[PHP 3, PHP 4]

Retourne l'argument sans les espaces de fin de chaîne.

Exemple avec chop()

```
<?php
$trimmed = chop($line);
?>
```

Note : [chop\(\)](#) diffère de sa cousine du Perl **chop**, qui supprime le dernier caractère de la chaîne. Voir aussi [trim\(\)](#), [ltrim\(\)](#), [rtrim\(\)](#), et [chop\(\)](#).

10.69.5 chr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [chr](#) (int *ascii*)
[PHP 3, PHP 4]

[chr\(\)](#) retourne le caractère de code ASCII *ascii*.

Exemple avec chr()

```
<?php
$str .= chr(27); /* ajoute un échappement à la fin de la chaîne $str */
/* Généralement, ceci est plus efficace */
$str = sprintf("Cette chaîne se termine par un escape: %c", 27);
?>
```

[chr\(\)](#) est le contraire de [ord\(\)](#). Voir aussi [sprintf\(\)](#) avec le format de chaîne %c.

10.69.6 chunk_split

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [chunk_split](#) (string *string*, int *chunklen* , string *end*)
[PHP 3>= 3.0.6, PHP 4]

[chunk_split\(\)](#) permet de scinder une chaîne en plus petit morceaux, comme dans le cas de la conversion en [10.73.2 base64_encode](#) pour se conformer à la RFC 2045. Cette fonction insère une fin de chaîne *end* (par défaut "\r\n"), tous les *chunklen* (par défaut 76) caractères. La chaîne retournée est une nouvelle chaîne, et l'original n'est pas modifié.

Exemple avec chunk_split()

```
<?php
# formate $data avec la sémantique RFC 2045
$new_string = chunk_split(base64_encode($data));
?>
```

[chunk_split\(\)](#) est nettement plus rapide que [ereg_replace\(\)](#). Note : [chunk_split\(\)](#) a été ajoutée en 3.0.6.

10.69.7 convert_cyr_string

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [convert_cyr_string](#) (string *str*, string *from*, string *to*)

[PHP 3>= 3.0.6, PHP 4]

[convert_cyr_string\(\)](#) convertit la chaîne donnée depuis un alphabet cyrillique vers un autre. Les arguments *from* et *to* sont des caractères qui représentent la source et la destination. Les valeurs acceptées :

- k – koi8-r
- w – windows-1251
- i – iso8859-5
- a – x-cp866
- d – x-cp866
- m – x-mac-cyrillic

[10.69.8 count_chars](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [count_chars](#)(string *string*, *mode*)

[PHP 4 >= 4.0b4]

Compte le nombre d'occurrence de chaque octet (0..255) dans la chaîne *string* et le retourne de différente façon. L'option *mode* prend, par défaut, la valeur 0. Suivant le *mode*, [count_chars\(\)](#) retourne une des réponses suivante :

- 0 – un tableau avec l'octet comme clé, et la fréquence comme valeur.
- 1 – Identique à 0, mais seule les fréquences non nulles sont listées.
- 2 – Identique à 0, mais seule les fréquences nulles sont listées.
- 3 – une chaîne qui contient tous les octets utilisés.
- 4 – une chaîne contenant tous les octets non utilisés.

Note : Cette fonction a été ajoutée en PHP 4.0.

[10.69.9 crc32](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [crc32](#)(string *str*)

[PHP 4 >= 4.0.1]

[crc32\(\)](#) génère la somme de vérification de redondance cyclique (32-bit) de la chaîne *str*. Cette valeur sert généralement à vérifier l'intégrité de données transmises.

Voir aussi: [md5\(\)](#)

[10.69.10 crypt](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [crypt](#)(string *str*, string *salt*)

[PHP 3, PHP 4]

[crypt\(\)](#) va coder une chaîne en utilisant la méthode d'encryption du DES standard. Les arguments sont : la chaîne à encrypter, et un grain de sel qui servira de base pour l'encryption. Reportez vous au manuel Unix pour plus de détails.

Si le grain de sel n'est pas fourni, il sera automatiquement généré par PHP.

Certains systèmes d'exploitation acceptent plus d'un type d'encryption. En fait, le DES standard est parfois remplacé par une encryption MD5. Le type d'encryption est alors choisi en fonction du grain de sel. A l'installation, PHP détermine les possibilités de cryptage et décidera d'accepter d'autres grains de sel pour d'autres types d'encryption. Si le grain de sel n'est pas fourni, PHP générera alors un grain de 2 caractères, pour le DES standard, à moins que le système ne dispose de MD5 : dans ce cas, PHP générera un grain de sel pour MD, par défaut. PHP affecte la variable d'environnement CRYPT_SALT_LENGTH, à 2 si il utilise le DES standard, et à 12 si il utilise le MD5.

Si vous utilisez le grain de sel fourni, retenez bien que ce grain de sel est généré une seule fois. Si vous appelez cette fonction récursivement, cela aura un impact sur l'apparance et finalement la sécurité de votre cryptage.

L'encryption standard fournit le grain de sel dans les deux premiers octets du résultat de la fonction [crypt\(\)](#). Sur les systèmes qui supportent plusieurs méthodes d'encryption, les variables d'environnement suivantes sont mises à 0 ou à 1, en fonction de la disponibilité de la méthode :

- CRYPT_STD_DES – DES Standard avec 2-octets de SALT
- CRYPT_EXT_DES – DES étendu avec 9-octets SALT
- CRYPT_MD5 – MD5 avec 12-octets SALT commençant à \$1\$
- CRYPT_BLOWFISH – DES étendu avec 16-octets SALT commençant à \$2\$

Il n'y a pas d'algorithme de décryptage, étant donné que [crypt\(\)](#) est injective.

10.69.11 echo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[echo](#) (string *arg1*, string *argn...*)

Affiche tous les paramètres.

[echo\(\)](#) n'est pas une fonction à proprement parler, ce qui rend l'usage des parenthèses facultatives.

Exemple avec echo()

```
<?php
echo "Bonjour Monde";
echo "Cet echo se
répartis sur plusieurs lignes. Les nouvelles lignes
seront aussi affichées";
echo "et echo se\nrépartis sur plusieurs lignes. Les nouvelles lignes\nseront aussi affichées.";
?>
```

Note : *En fait, si vous voulez passer plus d'un paramètre à [echo\(\)](#), vous ne DEVEZ pas les placer entre parenthèses.*

Voir aussi : [print\(\)](#), [printf\(\)](#) et [flush\(\)](#).

10.69.12 explode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [explode](#) (string *separator*, string *string*, int *limit*)

[PHP 3, PHP 4]

Retourne un tableau qui contient les éléments de la chaîne, séparés par *separator*.

Exemple avec `explode()`

```
<?php
$pizza = "piece1 piece2 piece3 piece4 piece5 piece6";
$pieces = explode(" ", $pizza);
?>
```

Note : Le paramètre **limit** a été ajouté en PHP 4.0.1.

Note : Bien que [implode\(\)](#) accepte, pour des raisons historiques, les arguments dans un sens ou l'autre, [explode\(\)](#) ne le peut. Vous devez vous assurer que l'argument séparateur **separator** arrive avant l'argument de chaîne.

Voir aussi [split\(\)](#) et [implode\(\)](#).

10.69.13 get_html_translation_table

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [get_html_translation_table](#)(int *table*, int *quote_style*)

[PHP 4 >= 4.0b4]

[get_html_translation_table\(\)](#) retourne la table de traduction utilisée en interne par [htmlspecialchars\(\)](#) et [htmlentities\(\)](#). Il y a deux nouvelles définitions : (*html_entities*, *html_specialchars*) qui vous permettent de spécifier vos propres tables.

Exemple de table de traduction

```
<?php
$trans = get_html_translation_table(HTML_ENTITIES);
$str = "Hallo 62;Frau> Kr38;auml;mer";
$encoded = strtr($str, $trans);
?>
```

La variable \$encoded va contenir désormais : "Hallo ; Frau ; Kr>;mer".

[array_flip\(\)](#) est alors très efficace pour inverser la direction de traduction :

```
<?php
$trans = array_flip($trans);
$original = strtr($str, $trans);
?>
```

Le contenu de \$original sera : "Hallo >Frau> Krämer". Note : Cette fonction a été ajoutée en PHP 4.0.

Voir aussi : [htmlspecialchars\(\)](#), [htmlentities\(\)](#), [strtr\(\)](#) et [array_flip\(\)](#).

10.69.14 get_meta_tags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [get_meta_tags](#)(string *filename*, int *use_include_path*)

[PHP 3 >= 3.0.4, PHP 4]

Ouvre le fichier *filename* et l'analyse ligne par ligne, en recherchant les balises <meta>.

Exemple avec les balises méta

```
<?php
<meta name="author" content="name">
<meta name="tags" content="php3 documentation">
</head> <!-- parsing stops here -->
?>
```

(Faîtes bien attention aux fins de lignes. PHP utilise une fonction native pour analyser le fichier d'entrée, ce qui fait que les fichiers créés sous MacOS ne fonctionneront pas sous Unix).

Le nom d'une propriété devient sa clé, et la valeur devient la valeur dans le tableau associatif retourné, ce qui rend aisé la manipulation des informations. Les caractères spéciaux dans la nom de la propriété sont remplacés par des '_', le reste est converti en minuscules.

Mettre *use_include_path* à 1 forcera PHP à ouvrir les fichiers dans le chemin standard d'inclusion.

10.69.15 hebrev

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hebrev](#) (string *hebrew_text*, int *max_chars_per_line*)
[PHP 3, PHP 4]

Le paramètre optionnel *max_chars_per_line* indique le nombre maximum de caractères par ligne qui seront générés. La fonction essaie d'éviter les césures de mots.

Voir aussi [hebrevc\(\)](#)

10.69.16 hebrevc

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [hebrevc](#) (string *hebrew_text*, int *max_chars_per_line*)
[PHP 3, PHP 4]

[hebrevc\(\)](#) est similaire à [hebrev\(\)](#), au détail près qu'elle convertit les nouvelles lignes (\n) en "
\n". Le paramètre optionnel *max_chars_per_line* indique le nombre maximum de caractères par ligne qui seront générés. La fonction essaie d'éviter les césures de mots.

Voir aussi [hebrev\(\)](#)

10.69.17 htmlentities

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [htmlentities](#) (string *string*, int *quote_style*)
[PHP 3, PHP 4]

Cette fonction est identique à [htmlspecialchars\(\)](#) en tous points, sauf que tous les caractères qui ont une entité équivalente en HTML sont remplacés par ces entités. Comme [htmlspecialchars\(\)](#), elle prend un argument optionnel qui indique ce qui doit être fait avec les guillemets simples et doubles. ENT_COMPAT (par défaut) convertira les guillemets doubles, et ignorera les guillemets simples. ENT_QUOTES convertira les deux types de guillemets et ENT_NOQUOTES les ignorera tous les deux.

Actuellement, le jeu de caractères ISO-8859-1 est utilisé. Notez que l'argument optionnel a été ajouté PHP

3.0.17 et PHP 4.0.3.

Voir aussi [htmlspecialchars\(\)](#) et [nl2br\(\)](#).

10.69.18 htmlspecialchars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [htmlspecialchars\(\)](#) (string *string*, int *quote_style*)

[PHP 3, PHP 4]

Certains caractères ont une valeur avec HTML, et doivent être remplacés par des balises HTML pour conserver leur valeur. Cette fonction retourne une chaîne avec tous les caractères sensibles remplacés par leur équivalent.

Cette fonction est utile pour empêcher un utilisateur de fournir un texte avec un sens HTML, comme dans un livre d'or.

[htmlspecialchars\(\)](#) est pratique pour éviter que les textes fournis par les utilisateurs contiennent des balises HTML, comme dans le cas d'un livre d'or ou d'une tribune. Cette fonction prend un argument optionnel qui indique ce qui doit être fait avec les guillemets simples et doubles. ENT_COMPAT (par défaut) convertira les guillemets doubles, et ignorera les guillemets simples. ENT_QUOTES convertira les deux types de guillemets et ENT_NOQUOTES les ignorera tous les deux.

Actuellement, PHP remplace les valeurs suivantes :

- '&' (et commercial) devient '&'
- '"' (guillemet double) devient '"'; si ENT_NOQUOTES n'est pas actif
- "'" (guillemet simple) devient '''; si ENT_QUOTES est actif
- '<' (inférieur à) devient '<'
- '>' (supérieur à) devient '>'

Exemple avec htmlspecialchars()

```
<?php
$new = htmlspecialchars("<a href='test'>Test</A>", ENT_QUOTES);
?>
```

Notez bien que cette fonction ne fait aucun autre remplacement que ceux listés ci-dessus. Pour une traduction complète de toutes les balises, reportez vous à [htmlentities\(\)](#). Notez que l'argument optionnel a été ajouté PHP 3.0.17 et PHP 4.0.3.

Voir aussi [htmlentities\(\)](#) et [nl2br\(\)](#).

10.69.19 implode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [implode\(\)](#) (string *glue*, array *pieces*)

[PHP 3, PHP 4]

Retourne une chaîne constituée de tous les éléments du tableau, pris dans l'ordre, transformés en chaîne, et séparés par *glue*.

Exemple avec implode()

```
<?php
$colon_separated = implode(":", $array);
?>
```

Note : Pour des raisons historiques, [implode\(\)](#) accepte ses arguments dans l'un ou l'autre sens. Par cohérence avec la fonction [explode\(\)](#), il est plus clair d'utiliser l'ordre des arguments tel que documenté. Voir aussi [explode\(\)](#), [join\(\)](#), et [split\(\)](#).

10.69.20 join

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [join](#) (string *glue*, array *pieces*)
[PHP 3, PHP 4]

[join\(\)](#) est un alias de [implode\(\)](#), et lui est identique en tous points. Voir aussi [explode\(\)](#), [implode\(\)](#), et [split\(\)](#).

10.69.21 levenshtein

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [levenshtein](#) (string *str1*, string *str2*) int [levenshtein](#) |string *str1*, string *str2*, int *cost_ins*, int *cost_rep*, int *cost_del*| int [levenshtein](#) |string *str1*, string *str2*, fonction *cost*
[PHP 3 >= 3.0.17, PHP 4 >= 4.0.1]

[levenshtein\(\)](#) retourne la distance Levenshtein entre les deux chaînes *str1* et *str2* ou -1 si un des arguments excède la limite de 255 caractères.

La distance Levenshtein est définie comme le nombre minimal de caractères qu'il faut remplacer, insérer ou effacer pour transformer la chaîne *str1* en *str2*. La complexité de l'algorithme est en $O(m*n)$, où n et m sont les longueurs respectives de *str1* et *str2* (ceci est plutôt un bon résultat, comparé à [similar_text\(\)](#), qui est en $O(\max(n,m)**3)$, mais cela reste coûteux en terme de ressources).

Dans sa forme la plus simple, la fonction va prendre uniquement deux chaînes en paramètres, et calculer uniquement le nombre d'insertion, remplacement et effacement nécessaire pour transformer la chaîne *str1* en *str2*.

Une variante prend trois paramètres additionnels, qui définissent le coût des insertions, des remplacements et des effacements. C'est une version plus générale et plus souple que la version simple, mais qui n'est pas aussi efficace.

La troisième variante n'est pas encore implémentée. Elle est encore plus générale, et plus souple, mais plus lente. Elle appellera une fonction utilisateur qui déterminera le coût de chaque opération.

La fonction utilisateur sera appelée avec les arguments suivants :

- opération à appliquer : 'I', 'R' or 'D'
- caractère courant de la chaîne *str1*
- caractère courant de la chaîne *str2*
- position courante de la chaîne *str1*
- position courante de la chaîne *str2*
- caractères restants dans la chaîne *str1*
- caractères restants dans la chaîne *str2*

La fonction utilisateur doit retourner un entier positif, qui décrira le cout de cette opération particulière. Elle peut ne prendre en compte que certains arguments, et non leur totalité.

L'utilisation d'une fonction utilisateur permet de prendre en compte la différence entre certains caractères, ou leur contexte pour déterminer le coût d'une opération d'insertion, remplacement ou effacement. Elle accroît la charge de calcul demandée au CPU, et annule l'optimisation des autres variantes.

Voir aussi [soundex\(\)](#), [similar_text\(\)](#) et [metaphone\(\)](#).

10.69.22 localeconv

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [localeconv](#)

[localeconv\(\)](#) retourne un tableau associatif contenant les informations locales de format monétaire et numérique utilisé par le serveur.

[localeconv\(\)](#) retourne les informations à partir des données locales, comme définies par [setlocale\(\)](#). Le tableau associatif retourné contient les entrées suivantes :

Index	Description
decimal_point	Séparateur décimal
thousands_sep	Séparateur de milliers
grouping	Tableau contenant les groupages numériques
int_curr_symbol	Symbole monétaire international (i.e. FRF)
currency_symbol	Symbole monétaire local (i.e. F)
mon_decimal_point	Séparateur décimal monétaire
mon_thousands_sep	Séparateur de milliers monétaires
mon_grouping	Tableau contenant les groupages numériques monétaires
positive_sign	Signe des valeurs positives
negative_sign	Signe des valeurs négatives
int_frac_digits	Nombre de chiffres décimaux international
frac_digits	Nombre de chiffres décimaux locaux
p_cs_precedes	TRUE si currency_symbol précède une valeur positive, FALSE s'il lui succède
p_sep_by_space	TRUE si un espace sépare currency_symbol d'une valeur positive, FALSE sinon

n_cs_precedes	TRUE si currency_symbol précède une valeur négative, FALSE s'il lui succède
n_sep_by_space	TRUE si un espace sépare currency_symbol d'une valeur négative, FALSE sinon
p_sign_posn	
0	
Des parenthèses entourent la quantité est currency_symbol	
1	
Le signe précède la quantité et currency_symbol	
2	
Le signe suit la quantité et currency_symbol	
3	
Le signe précède immédiatement currency_symbol	
4	
Le signe suit immédiatement currency_symbol @tab	
n_sign_posn	
0	
Des parenthèses entourent la quantité est currency_symbol	
1	
Le signe précède la quantité et currency_symbol	
2	
Le signe suit la quantité et currency_symbol	
3	
Le signe précède immédiatement currency_symbol	
4	
Le signe suit immédiatement currency_symbol @tab	

Le champs de groupage contient un tableau qui définit comment les chiffres doivent être regroupés. Par exemple, ce champs pour le dollar américain contient un tableau de deux éléments (3 et 3). Les éléments sont classés de gauche à droite. Si un des éléments vaut CHAR_MAX, les groupages ne sont plus effectués. Si un éléments vaut 0, la valeur du précédent doit être utilisée.

Exemple avec localeconv()

```
setlocale(LC_ALL, "en_US");
$locale_info = localeconv();
echo "<PRE>\n";
echo "-----\n";
```

```

echo " Informations monétaires pour le serveur local: \n";
echo "-----\n\n";
echo "int_curr_symbol:    {$locale_info["int_curr_symbol"]}\n";
echo "currency_symbol:    {$locale_info["currency_symbol"]}\n";
echo "mon_decimal_point:  {$locale_info["mon_decimal_point"]}\n";
echo "mon_thousands_sep:  {$locale_info["mon_thousands_sep"]}\n";
echo "positive_sign:      {$locale_info["positive_sign"]}\n";
echo "negative_sign:      {$locale_info["negative_sign"]}\n";
echo "int_frac_digits:    {$locale_info["int_frac_digits"]}\n";
echo "frac_digits:        {$locale_info["frac_digits"]}\n";
echo "p_cs_precedes:       {$locale_info["p_cs_precedes"]}\n";
echo "p_sep_by_space:      {$locale_info["p_sep_by_space"]}\n";
echo "n_cs_precedes:       {$locale_info["n_cs_precedes"]}\n";
echo "n_sep_by_space:      {$locale_info["n_sep_by_space"]}\n";
echo "p_sign_posn:         {$locale_info["p_sign_posn"]}\n";
echo "n_sign_posn:         {$locale_info["n_sign_posn"]}\n";
echo "</PRE>\n";

```

La constante CHAR_MAX est aussi définie ci-dessus.

Note : Ajouté en PHP 4.0.5

Voir aussi : [setlocale\(\)](#).

10.69.23 ltrim

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ltrim](#) (string *str*)

[PHP 3, PHP 4]

Cette fonction enlève les caractères blancs placés au début d'une chaîne et retourne la chaîne raccourcie. Les caractères blancs sont : "\n", "\r", "\t", "\v", "\0", et " ".

Voir aussi [chop\(\)](#) et [trim\(\)](#).

10.69.24 md5

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [md5](#) (string *str*)

[PHP 3, PHP 4]

Crypte la chaîne *str* en utilisant la méthode MD5 (voir [RSA Data Security, Inc. MD5 Message-Digest Algorithm](#)).

10.69.25 metaphone

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [metaphone](#) (string *str*)

[PHP 4 >= 4.0b4]

[metaphone\(\)](#) calcule la clé métaphone de la chaîne *str*.

Similairement à [soundex\(\)](#), [metaphone](#) crée une clé similaire pour des sons proches. C'est une fonction plus précise que [soundex\(\)](#) car elle prend en compte les règles basiques de la prononciation en anglais. Les clés métaphones sont de longueur variable.

Metaphone a été développé par Lawrence Philips <lphilips@verity.com>. Elle est décrit dans ["Practical

Algorithms for Programmers", Binstock & Rex, Addison Wesley, 1995]. Note : *Cette fonction a été ajoutée en PHP 4.0.*

10.69.26 nl2br

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [nl2br](#)(string *string*)
[PHP 3, PHP 4]

[nl2br\(\)](#) retourne la chaîne *string* dont toutes les lignes ont été remplacées par '
'.
Voir aussi [htmlspecialchars\(\)](#) et [htmlentities\(\)](#).

10.69.27 ord

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [ord](#)(string *string*)
[PHP 3, PHP 4]

Retourne la valeur ASCII du premier caractère de la chaîne *string*. Cette fonction est le contraire de [chr\(\)](#).
Exemple avec ord()

```
<?php
if (ord($str) == 10) {
echo "Le premier caractère de \"$str\" est un retour chariot.\n";
}
?>
```

Voir aussi [chr\(\)](#).

10.69.28 parse_str

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [parse_str](#)(string *str*, array *arr*)
[PHP 3, PHP 4]

Analyse la chaîne *str* comme si c'était une chaîne passée par URL, et affecte les variables qu'elle y trouve.

Utilisation de parse_str()

```
<?php
$str = "first=valeur&second[]=ceci+marche&second[]=encore";
parse_str($str);
echo $first;          /* affiche "valeur" */
echo $second[0];      /* affiche "ceci marche" */
echo $second[1];      /* affiche "encore" */
?>
```

10.69.29 print

[\[Notes en ligne\]](#) [\[Exemples\]](#)

print (string *arg*)

print() affiche *arg*.

Voir aussi : [echo\(\)](#), [printf\(\)](#), et [flush\(\)](#).

10.69.30 printf

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int printf (string *format*, mixed *args*...)

[PHP 3, PHP 4]

Affiche les arguments en fonction du *format*. Ce format est décrit en détails dans la documentation de [sprintf\(\)](#).

Voir aussi : [print\(\)](#), [sprintf\(\)](#), [sscanf\(\)](#), [fscanf\(\)](#), et [flush\(\)](#).

10.69.31 quoted_printable_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string quoted_printable_decode (string *str*)

[PHP 3>= 3.0.6, PHP 4]

quoted_printable_decode() retourne une chaîne 8-bit résultant du décodage de la chaîne *str*.

quoted_printable_decode() est similaire à [imap_qprint\(\)](#), hormis le fait qu'elle ne requiert pas le module *IMAP*.

10.69.32 quotemeta

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string quotemeta (string *str*)

[PHP 3, PHP 4]

quotemeta() retourne une version de la chaîne *str*, avec un antislash (\) devant tous les caractères de la liste ci-dessous : @example . \ + * ? [^] (\$) .

Voir aussi [addslashes\(\)](#), [htmlentities\(\)](#), [htmlspecialchars\(\)](#), [nl2br\(\)](#) et [stripslashes\(\)](#).

10.69.33 rtrim

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string rtrim (string *str*)

[PHP 3, PHP 4]

rtrim() la chaîne *str*, débarrassée de ses espaces terminaux, y compris les nouvelles lignes. Cette fonction est un alias de [chop\(\)](#).

Exemple avec rtrim() example

```
<?php
$trimmed = rtrim($line);
?>
```

Voir aussi [trim\(\)](#), [ltrim\(\)](#), et [rtrim\(\)](#).

10.69.34 sscanf

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [sscanf\(\)](#) (string *str*, string *format*, string *var1*...)
[PHP 4 >= 4.0.1]

[sscanf\(\)](#) est le complémentaire de [printf\(\)](#). [sscanf\(\)](#) lit les données de la chaîne *str* et interprète son contenu en fonction du format *format*. Si seulement deux paramètres sont passés à cette fonction, les valeurs obtenues seront retournées sous forme d'un tableau.

Exemple avec [sscanf\(\)](#)

```
<?php
// lecture d'un numéro de série
$serial = sscanf("SN/2350001", "SN/%d");
// et la date de fabrication
$mandate = "January 01 2000";
list($month, $day, $year) = sscanf($mandate, "%s %d %d");
echo "Le produit $serial a été fabriqué le: $year-".substr($month, 0, 3)."- $day\n";
?>
```

Si les paramètres optionnels sont passés, [sscanf\(\)](#) retournera le nombre de valeurs assignés. Les options doivent être passés par référence.

Utilisation des options avec [sscanf\(\)](#)

```
<?php
// Lecture des informations d'auteur, et génération
// d'une entrée DocBook
$auth = "24\tVictor Hugo";
$n = sscanf($auth, "%d\t%s %s", &$id, &$first, &$last);
echo "<auteur id='$id'>
    <Prénom>$first</firstname>
    <Nom>$last</surname>
</auteur>\n";
?>
```

Voir aussi: [fscanf\(\)](#), [printf\(\)](#), et [sprintf\(\)](#).

10.69.35 setlocale

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [setlocale\(\)](#) (string *category*, string *locale*)
[PHP 3, PHP 4]

category est une chaîne qui spécifie la catégorie de fonction qui va être affectée par les informations locales :

- LC_ALL Toutes les fonctions ci-dessous
- LC_COLLATE pour les comparaisons de chaîne (voir [strcoll\(\)](#))
- LC_CTYPE pour la classification de caractères et les conversions, par exemple [strtoupper\(\)](#)
- LC_MONETARY pour localeconv() – (en cours d'implémentation)
- LC_NUMERIC pour les séparateurs décimaux
- LC_TIME pour le format des dates et heures date avec [strftime\(\)](#)

Si *locale* est une chaîne vide (""), les noms locaux prendront la valeur des variables d'environnement de même nom, ou à partir de "LANG".

Si *locale* vaut zéro ou "0", la valeur reste inchangée, mais l'état courant est retourné.

[setlocale\(\)](#) retourne la valeur courante, ou FALSE si la fonctionnalité n'est pas encore implémentée pour la plateforme. Une catégorie invalide provoque une alerte.

[10.69.36 similar_text](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [similar_text](#)(string *first*, string *second*, double *percent*)

[PHP 3 >= 3.0.7, PHP 4 >= 4.0b2]

Cette fonction calcule la similarité entre deux chaînes, comme décrit par Oliver [1993]. Notez que cette implémentation n'utilise pas une pile, comme dans le pseudo-code d'Oliver, mais un appel récursif qui accélère parfois l'exécution. Notez aussi que la complexité de cet algorithme est en $O(N^3)$ avec N la taille de la plus grande chaîne.

En passant une référence comme troisième argument, [similar_text\(\)](#) va calculer le pourcentage de similarité. Il retourne le nombre de caractères correspondant l'un à l'autre, d'une chaîne à l'autre.

[10.69.37 soundex](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [soundex](#)(string *str*)

[PHP 3, PHP 4]

[soundex\(\)](#) calcule la valeur soundex de *str*.

Une valeur Soundex est telle que deux mots prononcés de la même façon auront des valeurs Soundex identiques. Cela permet d'effectuer des recherches dans les bases de données, si vous connaissez la prononciation mais pas l'orthographe. Cette fonction retourne une chaîne de 4 caractères, commençant par une lettre.

Cette fonction particulière a été décrite par Donald Knuth dans "The Art Of Computer Programming, vol. 3: Sorting And Searching", Addison-Wesley (1973), pp. 391–392.

Exemple avec Soundex

```
<?php
soundex("Euler") == soundex("Ellery") == 'E460';
soundex("Gauss") == soundex("Ghosh") == 'G200';
soundex("Knuth") == soundex("Kant") == 'H416';
soundex("Lloyd") == soundex("Ladd") == 'L300';
soundex("Lukasiewicz") == soundex("Lissajous") == 'L222';
?>
```

10.69.38 [sprintf](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sprintf](#)(string *format*, mixed *args...*)
[PHP 3, PHP 4]

[sprintf\(\)](#) retourne une chaîne formatée avec le format *format*.

La chaîne de format est composée de 0 ou plus directives : généralement des caractères qui sont recopiés tels quel (hormis %), et des spécifications, chacune d'elle disposant de son propre paramètre. Cela s'applique à [sprintf\(\)](#) et [printf\(\)](#).

Chaque conversion consiste, en un signe pourcentage (%), suivi d'un ou plusieurs éléments parmi ceux-ci :

1. Une option de remplissage, qui indique quel caractère sera utilisé pour le remplissage, et la taille finale de la chaîne. Le caractère de remplissage peut être un espace ou le caractère zéro (0). La valeur par défaut est l'espace. Une autre valeur peut être spécifiée en la préfixant par un guillemet simple ('). Voir les exemples plus loin.
2. Un argument optionnel *alignment specifier* qui indique que le résultat doit être justifié à droite ou à gauche. Par défaut, il est justifié à gauche. Le caractère – signifie : justification à droite.
3. Argument optionnel, *width specifier* indique le nombre minimum de caractères que la conversion devrait retourner.
4. Argument optionnel, *precision specifier* indique le nombre de chiffres utilisé pour afficher un nombre à virgule flottante. Cette option n'a d'effet que sur les nombres à virgule, double. (Une autre fonction pratique pour formater les nombres est : [number_format\(\)](#).)
5. *type specifier* indique le type de données passées en argument : Les types possibles sont :
 - ◆ % – un signe pourcentage : aucun argument nécessaire.
 - ◆ b – l'argument est traité comme un entier, et représenté comme un nombre binaire.
 - ◆ c – l'argument est traité comme un entier, et représenté comme un nombre ascii.
 - ◆ d – l'argument est traité comme un entier, et représenté comme un nombre décimal.
 - ◆ f – l'argument est traité comme un double, et représenté comme un nombre à virgule flottante.
 - ◆ o – l'argument est traité comme un entier, et représenté comme un nombre octal.
 - ◆ s – l'argument est traité tel quel, et représenté comme une chaîne.
 - ◆ x – l'argument est traité comme un entier, et représenté comme un nombre hexadécimal (en minuscules).
 - ◆ X – l'argument est traité comme un entier, et représenté comme un nombre hexadécimal (en majuscules).

Voir aussi : [printf\(\)](#), [sscanf\(\)](#), [fscanf\(\)](#), et [number_format\(\)](#).

Exemple avec [sprintf\(\)](#): complété avec des zéros

```
<?php
$isodate = sprintf("%04d-%02d-%02d", $year, $month, $day);
?>
```

Exemple avec [sprintf\(\)](#): format monétaire

```
<?php
$money1 = 68.75;
$money2 = 54.35;
$money = $money1 + $money2;
// echo $money affichera "123.1";
$formatted = sprintf("%01.2f", $money);
// echo $formatted affichera "123.10"
?>
```

10.69.39 strncasecmp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strncasecmp](#)(string *str1*, string *str2*, int *len*)

[PHP 4 >= 4.0.2]

[strncasecmp\(\)](#) est similaire à [strcasecmp\(\)](#), à la différence près qu'elle permet de limiter le nombre de caractères utilisés pour comparer *str1* et *str2*, avec le paramètre *len*. Si une des chaînes est plus courte que *len*, alors la longueur de cette chaîne sera utilisée pour effectuer la comparaison.

[strncasecmp\(\)](#) retourne < 0 si *str1* est plus petit que *str2*; > 0 si *str1* est plus grand que *str2*, et 0 si elles sont égales.

Voir aussi [ereg\(\)](#), [strcasecmp\(\)](#), [strcmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), et [strstr\(\)](#).

10.69.40 strcasecmp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcasecmp](#)(string *str1*, string *str2*)

[PHP 3 >= 3.0.2, PHP 4]

Retourne < 0 si *str1* est plus petit que *str2*; > 0 si *str1* est plus grand que *str2*, et 0 si ils sont égaux.

Exemple avec strcmp()

```
<?php
$var1 = "Bonjour";
$var2 = "bonjour";
if ( !strcasecmp($var1,$var2) ) {
echo '$var1 est égal à $var2, à la casse près.';
}
?>
```

Voir aussi [ereg\(\)](#), [strcmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), [strncasecmp\(\)](#) et [strstr\(\)](#).

10.69.41 strchr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strchr](#)(string *haystack*, string *needle*)

[PHP 3, PHP 4]

[strchr\(\)](#) est un alias de [strstr\(\)](#), et lui est identique en tous points.

10.69.42 strcmp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcmp](#)(string *str1*, string *str2*)

[PHP 3, PHP 4]

[strcmp\(\)](#) retourne < 0 si *str1* est plus petit que *str2*; > 0 si *str1* est plus grand que *str2*, et 0 si ils sont égaux. Notez bien que la comparaison est sensible à la casse.

Voir aussi [ereg\(\)](#), [strcasecmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), [strncmp\(\)](#), [strncasecmp\(\)](#) et [strstr\(\)](#).

10.69.43 strcoll

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcoll](#)(string *str1*, string *str2*)

[strcoll\(\)](#) retourne < 0 si *str1* est plus petit que *str2*; > 0 si *str1* est plus grand que *str2*, et 0 si elles sont égales. [strcoll\(\)](#) utilise la configuration locale pour effectuer les comparaisons. Si la configuration locale est : C ou POSIX, [strcoll\(\)](#) est équivalente à [strcmp\(\)](#).

Notez que cette comparaison est sensible à la casse, et que contrairement à [strcmp\(\)](#), [strcoll\(\)](#) n'est pas binaire.

Note : Ajoutée en PHP 4.0.5.

Voir aussi [ereg\(\)](#), [strcmp\(\)](#), [strcasecmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), [strncasecmp\(\)](#), [strncmp\(\)](#), [strstr\(\)](#), et [setlocale\(\)](#).

10.69.44 strcspn

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strcspn](#)(string *str1*, string *str2*)

[PHP 3>= 3.0.3, PHP 4]

[strcspn\(\)](#) retourne la longueur du premier segment de la chaîne *str1* qui ne contiennent pas aucun des caractères de la chaîne *str2*.

Voir aussi [strspn\(\)](#).

10.69.45 strip_tags

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strip_tags](#)(string *str*, string *allowable_tags*)

[PHP 3>= 3.0.8, PHP 4 >= 4.0b2]

[strip_tags\(\)](#) recherche et supprime toutes les balises HTML et PHP d'une chaîne. En cas de balises non fermées, ou de balises mal formées, elle génère une erreur. [strip_tags\(\)](#) utilise le même système que la fonction [fgetss\(\)](#).

Vous pouvez utiliser l'option *allowable_tags* pour spécifier les balises qui seront ignorées. Note : *allowable_tags* a été ajouté en PHP 3.0.13, et PHP 4.0B3.

10.69.46 stripslashes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [stripcslashes](#) (string *str*)

[PHP 4 >= 4.0b4]

[stripcslashes\(\)](#) retourne une chaîne dont les antislash ont été supprimés. Cette fonction reconnaît les \n, \r ..., et les représentations octales et hexadécimales utilisées en C. Note : [stripcslashes\(\)](#) a été ajouté dans PHP 4.0b3-dev.

Voir aussi [addslashes\(\)](#).

10.69.47 stripslashes

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [stripslashes](#) (string *str*)

[PHP 3, PHP 4]

[stripslashes\(\)](#) retourne une chaîne dont tous les slashes ont été supprimés. (\' devient ', ... et ainsi de suite). Les doubles antislash sont remplacés par des simples.

Voir aussi [addslashes\(\)](#).

10.69.48 strpos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strpos](#) (string *haystack*, string *needle*)

[PHP 3 >= 3.0.6, PHP 4]

[strpos\(\)](#) retourne tous les éléments de *haystack* à partir de la première occurrence de *needle*, jusqu'à la fin. *needle* et *haystack* sont examinés sans tenir compte de la casse.

Si *needle* n'est pas trouvé, retourne FALSE.

Si *needle* n'est pas une chaîne, elle est convertie en entier, est utilisée comme si elle était passée à [chr\(\)](#).

Voir aussi [strchr\(\)](#), [strrchr\(\)](#), [substr\(\)](#), et [ereg\(\)](#).

10.69.49 strlen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strlen](#) (string *str*)

[PHP 3, PHP 4]

[strlen\(\)](#) retourne la longueur de la chaîne *string*.

10.69.50 strncmp

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strncmp](#) (string *str1*, string *str2*)

[PHP 4 >= 4.0RC2]

[strnatcmp\(\)](#) implémente un algorithme de comparaison qui traite les chaînes alphanumériques comme un être humain : c'est ce qui est appelé l'"ordre naturel". Un exemple de la différence de traitement entre un tel algorithme et un algorithme de comparaison de chaîne (comme lorsqu'on utilise [strcmp\(\)](#)) est illustré ci dessous :

```
<?php
$arr1 = $arr2 = array("img12.png", "img10.png", "img2.png", "img1.png");
echo "Comparaison standard de chaînes\n";
usort($arr1, "strcmp");
print_r($arr1);
echo "\nComparaison de chaînes par ordre naturel\n";
usort($arr2, "strnatcmp");
print_r($arr2);
?>
```

L'exemple précédent affiche ceci : Comparaison standard de chaînes Array ([0] => img1.png [1] => img10.png [2] => img12.png [3] => img2.png) Comparaison de chaînes par ordre naturel Array ([0] => img1.png [1] => img2.png [2] => img10.png [3] => img12.png) Pour plus d'informations, reportez vous à Martin Pool [Natural Order String Comparison](#).

Comme les autres fonctions de comparaisons de chaînes, elle retourne une valeur < 0 si *str1* est plus petites que *str2*; > 0 si *str1* est plus grande que *str2*, et 0 si elles sont égales.

Notez que ces comparaisons sont sensibles à la casse.

Voir aussi [ereg\(\)](#), [strcasecmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), [strcmp\(\)](#), [strncmp\(\)](#), [strnatcasecmp\(\)](#), [strsr\(\)](#), [natsort\(\)](#), [strncasecmp\(\)](#) et [natcasesort\(\)](#).

[10.69.51 strnatcasecmp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strnatcasecmp](#)(string *str1*, string *str2*)
[PHP 4 >= 4.0RC2]

[strnatcasecmp\(\)](#) implémente un algorithme de comparaison qui traite les chaînes alphanumériques comme un être humain : c'est ce qui est appelé l'"ordre naturel". Pour plus d'informations, reportez vous à Martin Pool [Natural Order String Comparison](#).

Comme les autres fonctions de comparaisons de chaînes, elle retourne une valeur < 0 si *str1* est plus petites que *str2*; > 0 si *str1* est plus grande que *str2*, et 0 si elles sont égales.

Voir aussi [ereg\(\)](#), [strcasecmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), [strcmp\(\)](#), [strncmp\(\)](#), [strnatcmp\(\)](#), [strncasecmp\(\)](#) et [strsr\(\)](#).

[10.69.52 strncmp](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strncmp](#)(string *str1*, string *str2*, int *len*)
[PHP 4 >= 4.0b4]

[strncmp\(\)](#) est similaire à [strcmp\(\)](#), à la différence près que vous pouvez spécifier le nombre limite de caractères (*len*) utilisés pour faire la comparaison. Si l'une des chaînes est plus courte que *len*, alors cette longueur sera utilisée pour faire la comparaison.

Comme les autres fonctions de comparaisons de chaînes, elle retourne une valeur < 0 si *str1* est plus petites que *str2*; > 0 si *str1* est plus grande que *str2*, et 0 si elles sont égales. equal.

Notez que la comparaison est sensible à la casse.

Voir aussi [ereg\(\)](#), [strcasecmp\(\)](#), [substr\(\)](#), [stristr\(\)](#), [strcmp\(\)](#), [strncasecmp\(\)](#) et [strstr\(\)](#).

10.69.53 str_pad

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [str_pad](#)(string *input*, int *pad_length*, string *pad_string*, int *pad_type*)
[PHP 4 >= 4.0.1]

[str_pad\(\)](#) complète la chaîne *input* à droite, à gauche ou dans les deux directions, avec *pad_string* jusqu'à la taille de *pad_length*. Si *pad_string* n'est pas fourni, *input* est complété avec des espaces. Sinon, il est complété avec *pad_string*.

pad_type peut prendre les valeurs de STR_PAD_RIGHT, STR_PAD_LEFT, ou STR_PAD_BOTH. Si *pad_type* n'est pas spécifiée, cela vaut STR_PAD_RIGHT.

Si *pad_length* est négative ou inférieure à la taille courante de la chaîne *input*, aucun complément n'est ajouté.

Exemple avec str_pad()

```
<?php
$input = "Paris";
print str_pad($input, 10);                // produces "Paris      "
print str_pad($input, 10, "-=", STR_PAD_LEFT); // produces "-==--Paris"
print str_pad($input, 10, "_", STR_PAD_BOTH);  // produces "__Paris__"
?>
```

10.69.54 strpos

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [strpos](#)(string *haystack*, string *needle*, int *offset*)
[PHP 3, PHP 4]

[strpos\(\)](#) retourne la position numérique de la première occurrence de *needle* dans la chaîne *haystack*.

Contrairement à [strrpos\(\)](#), *needle* peut être une chaîne.

Si *needle* n'est pas trouvée, retourne FALSE. Note : *Il est facile de confondre la valeur de retour "caractère trouvé à la position 0 et "caractère introuvable". Voici comment faire la différence :*

```
<?php
// PHP 4.0b3 et plus récent :
$pos = strpos("b", $mystring);
if ($pos === FALSE) { // note: trois signes égal
    // non trouvé

// versions plus anciennes que 4.0b3:
$pos = strpos("b", $mystring);
if (is_string($pos) && !$pos) {
    // non trouvé
}
?>
```

}

Si *needle* n'est pas une chaîne, elle est convertie en entier, et utilisée comme la valeur ASCII d'un caractère. L'argument optionnel *offset* permet de préciser le caractère à partir duquel chercher, dans *haystack*. La position doit être relative au début de la chaîne *haystack*. Voir aussi [strrpos\(\)](#), [strrchr\(\)](#), [substr\(\)](#), [stristr\(\)](#) et [strstr\(\)](#).

10.69.55 strrchr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strrchr](#)(string *haystack*, string *needle*)
[PHP 3, PHP 4]

[strrchr\(\)](#) retourne la partie de la chaîne *haystack* qui commence à la dernière occurrence de *needle* et va jusqu'à la fin de la chaîne *haystack*.

Retourne FALSE si *needle* n'est pas trouvé.

Si *needle* contient plus d'un caractère, les autres sont ignorés.

Si *needle* n'est pas une chaîne, il est converti en un entier, et utilisé comme valeur ordinaire.

Exemple avec strrchr()

```
<?php
// lis le dernier repertoire de $PATH
$dir = substr(strrchr($PATH, ":"), 1);
// lis tout après la dernière ligne
$text = "Line 1\nLine 2\nLine 3";
$last = substr(strrchr($text, 10), 1 );
?>
```

Voir aussi [substr\(\)](#), [stristr\(\)](#), et [strstr\(\)](#).

10.69.56 str_repeat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [str_repeat](#)(string *input*, int *multiplier*)
[PHP 4 >= 4.0b4]

[str_repeat\(\)](#) retourne *input_str* répétées *multiplier* fois. *multiplier* doit être supérieur à 0.

Exemple avec str_repeat()

```
<?php
echo str_repeat("-", 10);
?>
```

Cet exemple affichera "-----".

Note : Cette fonction a été ajoutée en PHP 4.0.

10.69.57 strrev

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strrev](#)(string *string*)

[PHP 3, PHP 4]

[strrev\(\)](#) retourne *string*, après avoir changé l'ordre des caractères.

[10.69.58 strrpos](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)int [strrpos](#)(string *haystack*, char *needle*)

[PHP 3, PHP 4]

[strrpos\(\)](#) retourne la position numérique de la dernière occurrence de *needle* dans la chaîne *haystack*.[strrpos\(\)](#) ne peut accepter qu'un seul caractère.Si *needle* n'est pas trouvé, retourne FALSE.Note : *Il est facile de confondre la valeur de retour "caractère trouvé à la position 0 et "caractère introuvable". Voici comment faire la différence :*

```
<?php
// PHP 4.0b3 et plus récent :
$pos = strrpos("b", $mystring);
if ($pos === FALSE) { // note: trois égal signes
    // non trouvé

// versions plus anciennes que 4.0b3:
$pos = strrpos("b", $mystring);
if (is_string($pos) 38; !$pos) {
    // non trouvé
}
?>
```

} Si *needle* n'est pas une chaîne, elle est convertie en entier, et utilisée comme la valeur ASCII d'un caractère.Voir aussi [strpos\(\)](#), [strchr\(\)](#), [substr\(\)](#), [strpos\(\)](#) et [strstr\(\)](#).

[10.69.59 strstr](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)int [strstr](#)(string *str1*, string *str2*)

[PHP 3>= 3.0.3, PHP 4]

[strstr\(\)](#) retourne la longueur du premier segment de *str1* qui est constitué entièrement de caractères dans la chaîne *str2*.

```
<?php
strstr("42 est une réponse, quelle est la question ...", "1234567890");
?>
```

affichera "2".

Voir aussi [strcspn\(\)](#).

10.69.60 strstr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strstr](#)(string *haystack*, string *needle*)

[PHP 3, PHP 4]

[strstr\(\)](#) retourne toute la chaîne *haystack* à partir de la première occurrence de *needle*, jusqu'à la fin.

Si *needle* n'est pas trouvé, retourne FALSE.

Note : Notez que cette fonction est sensible à la casse : sinon, utilisez [stristr\(\)](#).

Si *needle* n'est pas une chaîne, elle est convertie en entier, et utilisée comme valeur ordinaire d'un caractère.

Exemple avec strstr()

```
<?php
$email = 'sterling@designmultimedia.com';
$domain = strstr($email, '');
print $domain; // affiche designmultimedia.com
?>
```

Voir aussi [stristr\(\)](#), [strchr\(\)](#), [substr\(\)](#), et [ereg\(\)](#).

10.69.61 strtok

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtok](#)(string *arg1*, string *arg2*)

[PHP 3, PHP 4]

[strtok\(\)](#) est utilisée pour morceller une chaîne. Pour cela, si vous avez une chaîne du type "ceci est une chaîne exemple", vous pouvez la morceller en mots, en utilisant ' ' comme délimiteur.

Exemple avec strtok()

```
<?php
$string = "ceci est une chaîne exemple";
$tok = strtok($string, " ");
while ($tok) {
    echo "Mot=$tok<br>";
    $tok = strtok(" ");
}
?>
```

Notez que seul, le premier appel à [strtok\(\)](#) utilise l'argument chaîne. Après, chaque appel à strtok ne requiert que le délimiteur à utiliser. Pour recommencer, vous pouvez simplement appeler [strtok\(\)](#) avec un nouvel argument, pour l'initialiser. Notez que vous pouvez mettre des délimiteurs multiples. La chaîne sera morcellée à chaque fois qu'on rencontrera un des délimiteurs.

Soyez prudents avec les délimiteurs qui sont égaux à "0". Cette valeur sera confondue avec FALSE.

Voir aussi [split\(\)](#) et [explode\(\)](#).

10.69.62 strtolower

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtolower](#) (string *str*)

[PHP 3, PHP 4]

[strtolower\(\)](#) retourne *string* avec tous les caractères alphabétiques en minuscule.

Notez que le caractère 'alphabétique' est déterminé par la table de caractères locale. Par exemple, dans la table des caractères par défaut du "C", des caractères tels que a–umlaut (ä) ne seront pas convertis.

Exemple avec strtolower()

```
<?php
$str = "Marie A Un Petit Agneau, Et Elle L'Adore";
$str = strtolower($str);
print $str; # Affiche : marie a un petit agneau, et elle l'adore
?>
```

Voir aussi [strtoupper\(\)](#) et [ucfirst\(\)](#).

10.69.63 strtoupper

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtoupper](#) (string *string*)

[PHP 3, PHP 4]

[strtoupper\(\)](#) retourne *string* avec tous ses caractères alphabétiques mis en majuscule.

Notez que le caractère 'alphabétique' est déterminé par la table de caractères locale. Par exemple, dans la table des caractères par défaut du "C", des caractères tels que a–umlaut (ä) ne seront pas convertis.

Exemple avec strtoupper()

```
<?php
$str = "Marie A Un Petit Agneau, Et Elle L'Adore";
$str = strtoupper($str);
print $str; # Affiche : MARIE A UN PETIT AGNEAU, ET ELLE L'ADORE
?>
```

Voir aussi [strtolower\(\)](#) et [ucfirst\(\)](#).

10.69.64 str_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [str_replace](#) (mixed *search*, mixed *replace*, mixed *subject*)

[PHP 3>= 3.0.6, PHP 4]

[str_replace\(\)](#) remplace toutes les occurrences de *search* dans *subject* par la chaîne *replace*. Si vous n'avez pas besoin de règles de remplacement sophistiquées, vous pouvez toujours utiliser [ereg_replace\(\)](#).

En PHP 4.0.5 et plus récent, chaque paramètre de [str_replace\(\)](#) peut être un tableau.

Si *subject* est un tableau, alors le remplacement est effectué pour chaque valeur de *subject*, et la valeur

retourné sera un tableau.

Si ***search*** et ***replace*** sont des tableaux, alors [str_replace\(\)](#) prend une valeur dans chaque tableau, et s'en sert pour chercher et remplacer dans ***subject***. Si ***replace*** a moins de valeurs que ***search***, des chaînes vides seront utilisées pour compléter le tableau ***replace***. Si ***search*** est un tableau et ***replace*** est une chaîne, alors la même chaîne de remplacement sera utilisée pour chaque valeur de ***search***. Le contraire n'aurait pas beaucoup de sens.

Exemple avec str_replace()

```
<?php
$bodytag = str_replace("%body%", "black", "<body text=%body%>");
?>
```

[str_replace\(\)](#) n'altère pas les données binaires.

Note : [str_replace\(\)](#) a été ajoutée dans PHP 3.0.6, mais était erronée jusqu'à PHP 3.0.8.

Voir aussi [ereg_replace\(\)](#), [preg_replace\(\)](#) et [strtr\(\)](#).

10.69.65 strtr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strtr](#)(string ***str***, string ***from***, string ***to***)

[PHP 3, PHP 4]

[strtr\(\)](#) travaille sur ***str***, remplaçant chaque occurrence de chaque caractère de la chaîne ***from*** correspondant à la chaîne ***to*** et retourne le résultat.

Si ***from*** et ***to*** sont de longueur différentes, les caractères en trop sont ignorés.

Exemple avec strtr()

```
<?php
$addr = strtr($addr, "AAÖ", "aao");
?>
```

[strtr\(\)](#) peut aussi être appelée avec deux arguments. Dans ce cas, elle se comporte différemment : ***from*** doit être un tableau associatif contenant des paires de chaînes, qui seront remplacées dans la chaîne source.

[strtr\(\)](#) recherchera toujours la chaîne la plus longue, et la remplacera en premier. Elle ne remplacera jamais une portion de chaîne qu'elle a déjà remplacé.

Exemples:

```
<?php
$trans = array("bonjour" => "salut", "salut" => "bonjour");
echo strtr("bonjour à tous, j'ai dit salut", $trans) . "\n";
?>
```

Cette exemple affichera : "salut à tous, j'ai dit bonjour",

Note : Travailler avec deux arguments a été ajouté dans PHP 4.0.

Voir aussi [ereg_replace\(\)](#).

10.69.66 substr

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [substr](#)(string *string*, int *start*, int *length*)

[PHP 3, PHP 4]

[substr\(\)](#) retourne une portion de *string*, spécifiée avec le début *start* et la longueur *length*.

Si *start* est positif, la chaîne retournée commencera au caractère *start* de la chaîne *string*. Par exemple, dans la chaîne 'abcdef', le caractère à la position 0 est 'a', le caractère à la position 2 est 'c', et ainsi de suite. Par exemple:

Exemples:

```
<?php
$rest = substr("abcdef", 1);    // retourne "bcdef"
$rest = substr("abcdef", 1, 3); // retourne "bcd"
?>
```

Si *start* est négatif, la chaîne retournée commencera au caractère *start* de la chaîne *string*, en partant de la fin.

Par exemple:

Exemples:

```
<?php
$rest = substr("abcdef", -1);    // retourne "f"
$rest = substr("abcdef", -2);    // retourne "ef"
$rest = substr("abcdef", -3, 1); // retourne "d"
?>
```

Si *length* est donné et positif, la chaîne retournée aura la longueur *length*. Si *length* est donné et négatif, la chaîne retournée aura la longueur *length*, en partant de la fin.

Exemples:

```
<?php
$rest = substr("abcdef", 1, -1); // retourne "bcde"
?>
```

Voir aussi [strchr\(\)](#) et [ereg\(\)](#).

10.69.67 substr_count

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [substr_count](#)(string *haystack*, string *needle*)

[PHP 4 >= 4.0RC2]

[substr_count\(\)](#) retourne le nombre de fois que *needle* apparaît dans *haystack*.

Exemple substr_count()

```
<?php
print substr_count("Ceci est un test", "es"); // affiche 2
```

?>

10.69.68 substr_replace

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [substr_replace](#) (string *string*, string *replacement*, int *start*, int *length*)
 [PHP 4 >= 4.0b4]

[substr_replace\(\)](#) effectue un remplacement dans la portion de *string* délimitée par le caractère *start* et de longueur optionnelle *length*. Le remplacement est fait avec la chaîne *replacement*. Le résultat est retourné. Si *start* est positif, le remplacement commencera au caractère *start*, dans la chaîne *string*. Si *start* est négative, le remplacement commencera au caractère *start* en partant de la fin de la chaîne *string*. Si *length* est donné et positif, la chaîne retournée aura la longueur *length*. Si *length* est donné et négatif, la chaîne retournée aura la longueur *length*, en partant de la fin. Par défaut, il prendra la valeur de `strlen(string)`; c'est à dire qu'il remplacera jusqu'à la fin de la chaîne *string*.

Exemple avec substr_replace()

```
<?php
$var = 'ABCDEFGH:/MNRPQR/';
echo "Original: $var<hr>\n";
/* Ces deux exemples remplacent tout $var avec 'bob'. */
echo substr_replace($var, 'bob', 0) . "<br>\n";
echo substr_replace($var, 'bob', 0, strlen($var)) . "<br>\n";
/* Insère 'bob' à gauche, du début de $var. */
echo substr_replace($var, 'bob', 0, 0) . "<br>\n";
/* Ces deux exemples remplacent 'MNRPQR' dans $var avec 'bob'. */
echo substr_replace($var, 'bob', 10, -1) . "<br>\n";
echo substr_replace($var, 'bob', -7, -1) . "<br>\n";
/* Efface 'MNRPQR' dans $var. */
echo substr_replace($var, '', 10, -1) . "<br>\n";
?>
```

Voir aussi [str_replace\(\)](#) et [substr\(\)](#).

Note : [substr_replace\(\)](#) a été ajoutée dans PHP 4.0.

10.69.69 trim

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [trim](#) (string *str*)
 [PHP 3, PHP 4]

Cette fonction retire les espaces blancs de début et de fin de chaîne, et retourne la chaîne nettoyée. Les espaces blancs sont : `"\n"`, `"\r"`, `"\t"`, `"\v"`, `"\0"`, et `" "` (espace).

Voir aussi [chop\(\)](#) et [ltrim\(\)](#).

10.69.70 ucfirst

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ucfirst](#) (string *str*)

[PHP 3, PHP 4]

Met le premier caractère d'une chaîne *str* en majuscule, si ce caractère est alphabétique.

Notez que le caractère 'alphabétique' est déterminé par la table de caractères locale. Par exemple, dans la table des caractères par défaut du "C", des caractères tels que a-umlaut (ä) ne seront pas convertis.

Exemple avec ucfirst()

```
<?php
$text = 'marie a un petit agneau, et l'adore.';
$text = ucfirst($text); // $text vaut : Marie a un petit agneau, et l'adore
?>
```

Voir aussi [strtoupper\(\)](#) et [strtolower\(\)](#).

10.69.71 ucwords

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [ucwords](#) (string *str*)

[PHP 3>= 3.0.3, PHP 4]

[ucwords\(\)](#) met le premier caractère de chaque mot de la chaîne *str* si ce caractère est une lettre.

Exemple avec ucwords()

```
<?php
$text = "marie a un petit agneau, et l'adore.";
$text = ucwords($text); // $text vaut : Marie A Un Petit Agneau, Et l'Adore.
?>
```

Note : La définition d'un mot est : une chaîne de caractères immédiatement après un caractère blanc (c'est à dire : espace, form-feed, nouvelle ligne, retour chariot, tabulation horizontale, et tabulation verticale).

Voir aussi [strtoupper\(\)](#), [strtolower\(\)](#) et [ucfirst\(\)](#).

10.69.72 wordwrap

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wordwrap](#) (string *str*, int *width* , string *break* , int *cut*)

[PHP 4 >= 4.0.2]

[wordwrap\(\)](#) ajoute la césure *str* au numéro de colonne *width*. La ligne est césurée avec la chaîne *break*.

[wordwrap\(\)](#) ajoute la césure automatiquement à la ligne 75 et utilise "\n" (nouvelle ligne) si *width* ou *break* sont omis.

Si *cut* vaut 1, la chaîne sera toujours coupée à la taille spécifiée. Dans ce cas, les mots qui dépasseront, seront coupés : voyez le second exemple.

Note : Le paramètre *cut* a été ajouté dans PHP 4.0.3.

Exemple wordwrap()

```
<?php
$text = "Maître corbeau jura, mais un peu tard, qu'on ne l'y prendrait plus.";
$newtext = wordwrap( $text, 20 );
echo "$newtext\n";
?>
```

Cet exemple va afficher :

Maître corbeau jura, mais un peu t ard, qu'on ne l'y pr endrait plus

Exemple avec wordwrap()

```
<?php
$text = "Un tres tres long mooooooooooooooooooooooooooooot.";
$newtext = wordwrap( $text, 8, "\n", 1);
echo "$newtext\n";
?>
```

Cet exemple va afficher

Un tres tres long mooooooo ooooooooo ooooooooo

Voir aussi [nl2br\(\)](#).

10.70 Shockwave Flash

[\[Notes en ligne\]](#)

PHP a la capacité de créer des animations Shockwave Flash grâce au module de Paul Haeberli : libswf module. Vous pouvez télécharger libswf à <http://reality.sgi.com/grafica/flash/>. Une fois que vous avez libswf, tout ce qui reste à faire est de configurer PHP avec `--with-swf[=DIR]` où DIR est le dossier qui accueille les dossiers de include et lib. Le dossier include doit contenir le fichier swf.h file et le dossier lib doit contenir le fichier libswf.a. Si vous décompressez la distribution de libswf, les deux fichiers seront dans le même dossier. Par conséquent, vous devrez les mettre dans le dossier ad hoc manuellement.

Une fois que vous avez réussi à installer PHP avec Shockwave Flash, vous pouvez créer des animations Flash avec PHP. Vous serez surpris du résultat. Essayez donc ceci :

Exemple SWF

```
<?php
swf_openfile ("test.swf", 256, 256, 30, 1, 1, 1);
swf_ortho2 (-100, 100, -100, 100);
swf_defineline (1, -70, 0, 70, 0, .2);
swf_definerect (4, 60, -10, 70, 0, 0);
swf_definerect (5, -60, 0, -70, 10, 0);
swf_addcolor (0, 0, 0, 0);
swf_definefont (10, "Mod");
swf_fontsize (5);
swf_fontslant (10);
swf_definetext (11, "This be Flash wit PHP!", 1);
swf_pushmatrix ();
swf_translate (-50, 80, 0);
swf_placeobject (11, 60);
swf_popmatrix ();
```

```

for ($i = 0; $i < 30; $i++) {
    $p = $i/(30-1);
    swf_pushmatrix ();
    swf_scale (1-($p*.9), 1, 1);
    swf_rotate (60*$p, 'z');
    swf_translate (20+20*$p, $p/1.5, 0);
    swf_rotate (270*$p, 'z');
    swf_addcolor ($p, 0, $p/1.2, -$p);
    swf_placeobject (1, 50);
    swf_placeobject (4, 50);
    swf_placeobject (5, 50);
    swf_popmatrix ();
    swf_showframe ();
}
for ($i = 0; $i < 30; $i++) {
    swf_removeobject (50);
    if (($i%4) == 0) {
        swf_showframe ();
    }
}
swf_startdoaction ();
swf_actionstop ();
swf_enddoaction ();
swf_closefile ();
?>

```

Cela va produire une animation, proche de [celle ci](#) (mais traduite en anglais).

Note : *Le support de Flash a été ajouté dans PHP 4.0RC2.*

La librairie libswf n'est pas disponible pour Windows : son développement a été stoppé, et les sources ne sont plus disponibles pour permettre le portage vers d'autres systèmes.

10.70.1 swf_openfile

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_openfile](#)(string *filename* , float *width* , float *height* , float *framerate* , float *r* , float *g* , float *b*)

[PHP 4 >= 4.0RC2]

[swf_openfile\(\)](#) crée un nouveau fichier *filename* de largeur *width*, et de hauteur *height*, à la vitesse de *framerate*, de couleur de fond RGB (*r*, *g*, *b*).

[swf_openfile\(\)](#) doit être la première fonction à appeler, sous peine d'erreur mémoire (segmentation fault). Si vous voulez envoyer votre production au client HTML, utilisez le nom de fichier "php://stdout" (le support de ceci est prévue pour la version 4.0.1 et ultérieur).

10.70.2 swf_closefile

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_closefile](#)(int *return_file*)

[PHP 4 >= 4.0RC2]

[swf_closefile\(\)](#) ferme le fichier courant, qui a été ouvert avec [swf_openfile\(\)](#). Si le paramètre *return_file* a été fourni, il contiendra le fichier SWF fermé.

Création d'un fichier Flash simple, basé sur une entrée de l'utilisateur, et sauvegarde dans une base.

```

<?php
// La variable $text est fournie par l'utilisateur
// Variables globales pour l'accès à la base de données
// utilisée dans la fonction wf_savedata()
$DBHOST = "localhost";
$DBUSER = "sterling";
$DBPASS = "secret";
swf_openfile ("php://stdout", 256, 256, 30, 1, 1, 1);
swf_definefont (10, "Ligon-Bold");
swf_fontsize (12);
swf_fontslant (10);
swf_definetext (11, $text, 1);
swf_pushmatrix ();
swf_translate (-50, 80, 0);
swf_placeobject (11, 60);
swf_popmatrix ();
swf_showframe ();
swf_startdoaction ();
swf_actionstop ();
swf_enddoaction ();
$data = swf_closefile (1);
$data ?
swf_savedata ($data) :
die ("Error could not save SWF file");
// void swf_savedata (string data)
// Sauve le fichier généré dans la base de données
// pour accès ultérieur
function swf_savedata ($data)
{
global $DBHOST,
        $DBUSER,
        $DBPASS;
        $dbh = @mysql_connect ($DBHOST, $DBUSER, $DBPASS);
if (!$dbh) {
die (sprintf ("Error [%d]: %s",
mysql_errno (), mysql_error ()));
}
        $stmt = "INSERT INTO swf_files (file) VALUES ('$data')";
        $sth = @mysql_query ($stmt, $dbh);
if (!$sth) {
die (sprintf ("Error [%d]: %s",
mysql_errno (), mysql_error ()));
}
        @mysql_free_result ($sth);
        @mysql_close ($dbh);
}
?>

```

10.70.3 swf_labelframe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_labelframe](#) (string *name*)

[PHP 4 >= 4.0RC2]

[swf_labelframe\(\)](#) donne le nom *name* au frame courant.

10.70.4 swf_showframe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_showframe](#)

[PHP 4 >= 4.0RC2]

[swf_showframe\(\)](#) affiche le frame courant.

10.70.5 swf_setframe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_setframe](#) (int *framenumber*)

[PHP 4 >= 4.0RC2]

[swf_setframe\(\)](#) selectionne le frame *framenumber* comme frame actif.

10.70.6 swf_getframe

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [swf_getframe](#)

[PHP 4 >= 4.0RC2]

[swf_getframe\(\)](#) retourne le numéro de frame courant.

10.70.7 swf_mulcolor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_mulcolor](#) (float *r* , float *g* , float *b* , float *a*)

[PHP 4 >= 4.0RC2]

[swf_mulcolor\(\)](#) fixe la valeur globale de multiplication (the global multiply color...) à la couleur *rgba*. Cette couleur est utilisée (implicitement) par [swf_placeobject\(\)](#), [swf_modifyobject\(\)](#) et [swf_addbuttonrecord\(\)](#). La couleur d'un objet sera multipliée par *rgba* lorsque l'objet est placé sur la scène.

Note : Les valeurs de *rgba* peuvent être positives ou négatives.

10.70.8 swf_addcolor

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_addcolor](#) (float *r* , float *g* , float *b* , float *a*)

[PHP 4 >= 4.0RC2]

[swf_addcolor\(\)](#) fixe la valeur globale de multiplication (the global multiply color...) à la couleur *rgba*. Cette couleur est utilisée (implicitement) par [swf_placeobject\(\)](#), [swf_modifyobject\(\)](#) et [swf_addbuttonrecord\(\)](#). La couleur d'un objet sera ajouté à *rgba* lorsque l'objet est placé sur la scène.

Note : Les valeurs de *rgba* peuvent être positives ou négatives.

[10.70.9 swf_placeobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_placeobject](#)(int *objid* , int *depth*)

[PHP 4 >= 4.0RC2]

[swf_placeobject\(\)](#) place l'objet *objid* dans le frame courant, à la profondeur *depth*. *objid* et *depth* doivent être compris entre 1 et 65535.

[swf_placeobject\(\)](#) utilise la couleur courante de multiplication (spécifiée par [swf_mulcolor\(\)](#)) et la couleur courante d'addition (spécifiée par [swf_addcolor\(\)](#)) pour colorer l'objet, et utilise la matrice courante pour positionner l'objet.

Note : *Le support des couleurs RGBA est complet.*

[10.70.10 swf_modifyobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_modifyobject](#)(int *depth* , int *how*)

[PHP 4 >= 4.0RC2]

[swf_modifyobject\(\)](#) modifie la position et/ou la couleur de l'objet situé à la profondeur de *depth*. L'argument *how* détermine ce qui doit être modifié. *how* peut prendre les valeurs de MOD_MATRIX, MOD_COLOR ou la combinaison des deux.

MOD_COLOR utilise la couleur courante de multiplication (spécifiée par [swf_mulcolor\(\)](#)) et la couleur courante d'addition (spécifiée par [swf_addcolor\(\)](#)) pour colorer l'objet, et MOD_MATRIX utilise la matrice courante pour positionner l'objet.

[10.70.11 swf_removeobject](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_removeobject](#)(int *depth*)

[PHP 4 >= 4.0RC2]

[swf_removeobject\(\)](#) enlève l'objet situé à la profondeur *depth* de la scène.

[10.70.12 swf_nextid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [swf_nextid](#)

[PHP 4 >= 4.0RC2]

[swf_nextid\(\)](#) retourne le prochain identifiant d'objet libre.

[10.70.13 swf_startdoaction](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startdoaction](#)

[PHP 4 >= 4.0RC2]

[swf_startdoaction\(\)](#) commence la description d'une liste d'actions pour la frame courante. Cette fonction doit être appelée avant que les actions ne soient définies pour le cadre courant.

[10.70.14 swf_actiongotoframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiongotoframe](#) (int *framenumbers*)
[PHP 4 >= 4.0RC2]

[swf_actiongotoframe\(\)](#) se déplace jusqu'au frame *framenumbers*, le joue, puis s'arrête.

[10.70.15 swf_actiongeturl](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiongeturl](#) (string *url* , string *target*)
[PHP 4 >= 4.0RC2]

[swf_actiongeturl\(\)](#) lit l'URL *url*, avec la destination *target*.

[10.70.16 swf_actionnextframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionnextframe](#)
[PHP 4 >= 4.0RC2]

[swf_actionnextframe\(\)](#) avance d'un frame le frame courant.

[10.70.17 swf_actionprevframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionprevframe](#)
[PHP 4 >= 4.0RC2]

[swf_actionprevframe\(\)](#) recule d'un frame le frame courant.

[10.70.18 swf_actionplay](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionplay](#)
[PHP 4 >= 4.0RC2]

[swf_actionplay\(\)](#) joue l'animation Flash à partir du frame courant.

[10.70.19 swf_actionstop](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionstop](#)

[PHP 4 >= 4.0RC2]

[swf_actionstop\(\)](#) arrête l'animation Flash au frame courant.

[10.70.20 swf_actiontogglequality](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiontogglequality](#)

[PHP 4 >= 4.0RC2]

[swf_actiontogglequality\(\)](#) modifie le niveau de qualité haut ou bas.

[10.70.21 swf_actionwaitforframe](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionwaitforframe](#) (int *framenumbers* , int *skipcount*)

[PHP 4 >= 4.0RC2]

[swf_actionwaitforframe\(\)](#) vérifie que le frame *framenumbers* a bien été chargé. Si ce n'est pas le cas, elle ignore les actions *skipcount*. Cela est très utile pour les séquences du type "Chargement...".

[10.70.22 swf_actionsettarget](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actionsettarget](#) (string *target*)

[PHP 4 >= 4.0RC2]

[swf_actionsettarget\(\)](#) fixe le contexte des actions. Vous pouvez utiliser cette fonction pour contrôler d'autres animations Flash qui seraient en fonctionnement.

[10.70.23 swf_actiongotolabel](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_actiongotolabel](#) (string *label*)

[PHP 4 >= 4.0RC2]

[swf_actiongotolabel\(\)](#) affiche le frame de nom *label*, puis stoppe.

[10.70.24 swf_enddoaction](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_enddoaction](#)

[PHP 4 >= 4.0RC2]

[swf_startdoaction\(\)](#) termine l'action courante, démarrée par [swf_startdoaction\(\)](#).

10.70.25 swf_defineline

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_defineline](#)(int *objid* , float *x1* , float *y1* , float *x2* , float *y2* , float *width*)
[PHP 4 >= 4.0RC2]

[swf_defineline\(\)](#) définit une ligne commençant aux coordonnées (*x1*, *y1*), et finissant au point de coordonnées (*x2*, *y2*). Elle aura la largeur de *width*.

10.70.26 swf_definerect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definerect](#)(int *objid* , float *x1* , float *y1* , float *x2* , float *y2* , float *width*)
[PHP 4 >= 4.0RC2]

[swf_definerect\(\)](#) définit un rectangle, de coin supérieur gauche aux coordonnées (*x1*, *y1*), et de coin inférieur droit aux coordonnées (*x2*, *y2*). L'épaisseur des bords est donnée par le paramètre *width*. *width*, 0.0 le rectangle sera rempli.

10.70.27 swf_definepoly

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definepoly](#)(int *objid* , array *coords* , int *npoints* , float *width*)
[PHP 4 >= 4.0.0]

[swf_definepoly\(\)](#) définit un polygone, dont les coordonnées des sommets sont placés dans le tableau *coords*. *npoints* est le nombre de points contenu dans le tableau *coords*. *width* est la largeur des bords du polygone. Si *width* vaut 0.0, le polygone sera rempli.

10.70.28 swf_startshape

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startshape](#)(int *objid*)
[PHP 4 >= 4.0RC2]

[swf_startshape\(\)](#) commence une forme complexe, qui sera repéré par l'identifiant d'objet *objid*.

10.70.29 swf_shapelinesolid

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapelinesolid](#)(float *r* , float *g* , float *b* , float *a* , float *width*)
[PHP 4 >= 4.0RC2]

[swf_shapelinesolid\(\)](#) permet de choisir le style de ligne, à savoir la couleur et la largeur. Si *width* vaut 0.0, les lignes ne seront pas dessinées.

[10.70.30 swf_shapefilloff](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefilloff](#)

[PHP 4 >= 4.0RC2]

[swf_shapefilloff\(\)](#) inactive le remplissage pour la forme courante.

[10.70.31 swf_shapefillsolid](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefillsolid](#)(float *r* , float *g* , float *b* , float *a*)

[PHP 4 >= 4.0RC2]

[swf_shapefillsolid\(\)](#) fixe la couleur pour le style courant de remplissage à *rgba*.

[10.70.32 swf_shapefillbitmapclip](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefillbitmapclip](#)(int *bitmapid*)

[PHP 4 >= 4.0RC2]

Choisi le mode de remplissage par texture : les espaces vides seront remplis avec la bitmap *bitmapid*.

[10.70.33 swf_shapefillbitmaptile](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapefillbitmaptile](#)(int *bitmapid*)

[PHP 4 >= 4.0RC2]

Choisi le mode de remplissage par texture : les espaces vides seront remplis avec la bitmap *bitmapid*, répétée autant de fois qu'il le faut (mode carrelage).

[10.70.34 swf_shapemoveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapemoveto](#)(float *x* , float *y*)

[PHP 4 >= 4.0RC2]

[swf_shapemoveto\(\)](#) fixe la position courante au point de coordonnées (*x*, *y*).

[10.70.35 swf_shapelineto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_shapelineto](#)(float *x* , float *y*)

[PHP 4 >= 4.0RC2]

[`swf_shapelineto\(\)`](#) dessine une ligne entre la position courante et le point de coordonnées (*x*, *y*). La position courante devient alors (*x*, *y*).

[10.70.36 swf_shapecurveto](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`swf_shapecurveto`](#)(float *x1* , float *y1* , float *x2* , float *y2*)
[PHP 4 >= 4.0RC2]

[`swf_shapecurveto\(\)`](#) dessine la courbe de Bézier quadratique entre les points de coordonnées (*x1* , *y1*) et (*x2*, *y2*). La position courante devient alors (*x2*, *y2*).

[10.70.37 swf_shapecurveto3](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`swf_shapecurveto3`](#)(float *x1* , float *y1* , float *x2* , float *y2* , float *x3* , float *y3*)
[PHP 4 >= 4.0RC2]

Dessine une courbe de Bézier cubique, en utilisant les points de coordonnées (*x1*, *y1*) et (*x2*, *y2*) comme points de contrôle et le point de coordonnées (*x3*, *y3*) comme point final. La position finale devient alors la position courante.

[10.70.38 swf_shapearc](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`swf_shapearc`](#)(float *x* , float *y* , float *r* , float *ang1* , float *ang2*)
[PHP 4 >= 4.0RC2]

[`swf_shapearc\(\)`](#) dessine un arc de cercle, depuis l'angle *ang1* jusqu'à l'angle *ang2*. Le centre du cercle est aux coordonnées (*x*, *y*), et de rayon *r*.

[10.70.39 swf_endshape](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`swf_endshape`](#)
[PHP 4 >= 4.0RC2]

[`swf_endshape\(\)`](#) complète la définition de la forme courante.

[10.70.40 swf_definefont](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [`swf_definefont`](#)(int *fontid* , string *fontname*)
[PHP 4 >= 4.0RC2]

[`swf_definefont\(\)`](#) définit la police *fontname* et lui affecte l'identifiant *fontid*. Cette police devient alors la police courante.

10.70.41 swf_setfont

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_setfont](#) (int *fontid*)

[PHP 4 >= 4.0RC2]

[swf_setfont\(\)](#) remplace la police courante par la police repérée par l'identifiant *fontid*.

10.70.42 swf_fontsize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_fontsize](#) (float *size*)

[PHP 4 >= 4.0RC2]

[swf_fontsize\(\)](#) remplace la taille de la police par la taille *size*.

10.70.43 swf_fontslant

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_fontslant](#) (float *slant*)

[PHP 4 >= 4.0RC2]

[swf_fontslant\(\)](#) fixe l'inclinaison de la police courante à *slant*. Les valeurs positives créent une inclinaison vers la droite, et les valeurs négatives, vers la gauche.

10.70.44 swf_fontracking

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_fontracking](#) (float *tracking*)

[PHP 4 >= 4.0RC2]

[swf_fontracking\(\)](#) change l'espacement, et lui affecte la valeur de *tracking*. Cette fonction sert à accroître l'espace entre les lettres et le texte. Les valeurs positives accroissent cet espace, et les valeurs négatives le réduisent.

10.70.45 swf_getfontinfo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [swf_getfontinfo](#)

[PHP 4 >= 4.0RC2]

[swf_getfontinfo\(\)](#) retourne la hauteur du A majuscule, et du x minuscule, dans un tableau associatif :

- Aheight – La hauteur du A majuscule, en pixels.
- xheight – La hauteur du x minuscule, en pixels.

10.70.46 swf_definetext

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definetext](#)(int *objid* , string *str* , int *docenter*)

[PHP 4 >= 4.0RC2]

[swf_definetext\(\)](#) définit la chaîne de texte *str*, en utilisant la police courante. *docenter* indique si la chaîne doit être centrée (valeur de 1), ou pas.

10.70.47 swf_textwidth

[\[Notes en ligne\]](#) [\[Exemples\]](#)

float [swf_textwidth](#)(string *str*)

[PHP 4 >= 4.0RC2]

[swf_textwidth\(\)](#) retourne la longueur de la chaîne *str*, en pixels, en utilisant la police courante.

10.70.48 swf_definebitmap

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_definebitmap](#)(int *objid* , string *image_name*)

[PHP 4 >= 4.0RC2]

[swf_definebitmap\(\)](#) définit une bitmap à partir d'une image au format GIF, JPEG, RGB ou FI. L'image sera convertie en Flash JPEG ou Flash color map.

10.70.49 swf_getbitmapinfo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [swf_getbitmapinfo](#)(int *bitmapid*)

[PHP 4 >= 4.0RC2]

[swf_getbitmapinfo\(\)](#) retourne un tableau d'informations sur l'image bitmap repérée par *bitmapid*. Le tableau a les éléments suivants :

- "size" – La taille en octets de l'image.
- "width" – La largeur en pixels de l'image.
- "height" – La hauteur en pixels de l'image.

10.70.50 swf_startsymbol

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startsymbol](#)(int *objid*)

[PHP 4 >= 4.0RC2]

[swf_startsymbol\(\)](#) définit un identifiant d'objet comme symbole. Les symboles sont des petites animations Flash qui peuvent être jouées simultanément. *objid* est l'identifiant d'objet que vous voulez définir comme symbole.

[10.70.51 swf_endsymbol](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_endsymbol](#)
[PHP 4 >= 4.0RC2]

[swf_endsymbol\(\)](#) termine la définition de symbole, qui a été commencée avec [swf_startsymbol\(\)](#).

[10.70.52 swf_startbutton](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_startbutton](#) (int *objid* , int *type*)
[PHP 4 >= 4.0RC2]

[swf_startbutton\(\)](#) commence la définition d'un bouton. *type* peut prendre les valeurs de TYPE_MENUBUTTON ou TYPE_PUSHBUTTON. La constante TYPE_MENUBUTTON permet au focus de traverser lorsque la souris est cliquée, alors que TYPE_PUSHBUTTON ne le permet pas.

[10.70.53 swf_addbuttonrecord](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_addbuttonrecord](#) (int *states* , int *shapeid* , int *depth*)
[PHP 4 >= 4.0RC2]

[swf_addbuttonrecord\(\)](#) permet de modifier les caractéristiques d'un bouton. *states*, définit les états du bouton autorisés : ce peut être : BSHitTest, BSDown, BSOVer ou BSUp. *shapeid* est l'apparence du bouton, c'est à dire l'objet qui représente le bouton. *depth* est la profondeur de placement du bouton, dans le frame courant.

Exemple avec swf_addbuttonrecord()

```
swf_startButton ($objid, TYPE_MENUBUTTON);
swf_addButtonRecord (BSDown|BSOver, $buttonImageId, 340);
swf_onCondition (MenuEnter);
swf_actionGetUrl ("http://www.designmultimedia.com", "_level1");
swf_onCondition (MenuExit);
swf_actionGetUrl ("", "_level1");
swf_endButton ();
```

[10.70.54 swf_oncondition](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_oncondition](#) (int *transition*)
[PHP 4 >= 4.0RC2]

[swf_oncondition\(\)](#) décrit une transition qui va déclencher une liste d'actions. Il y a plusieurs types de transition possibles, les suivantes sont destinées aux boutons de type TYPE_MENUBUTTON:

- IdletoOverUp
- OverUptoIdle
- OverUptoOverDown
- OverDowntoOverUp
- IdletoOverDown
- OutDowntoIdle
- MenuEnter (IdletoOverUp|IdletoOverDown)
- MenuExit (OverUptoIdle|OverDowntoIdle)

Pour les types TYPE_PUSHBUTTON voici les options :

- IdletoOverUp
- OverUptoIdle
- OverUptoOverDown
- OverDowntoOverUp
- OverDowntoOutDown
- OutDowntoOverDown
- OutDowntoIdle
- ButtonEnter (IdletoOverUp|OutDowntoOverDown)
- ButtonExit (OverUptoIdle|OverDowntoOutDown)

10.70.55 swf_endbutton

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_endbutton](#)
[PHP 4 >= 4.0RC2]

[swf_endbutton\(\)](#) termine la définition du bouton courant.

10.70.56 swf_viewport

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_viewport](#) (double *xmin* , double *xmax* , double *ymin* , double *ymax*)
[PHP 4 >= 4.0RC2]

[swf_viewport\(\)](#) sélectionne une nouvelle zone pour y dessiner ultérieurement. La zone est définie de *xmin* à *xmax* et de *ymin* à *ymax*. Si cette fonction n'est pas appelée, les valeurs par défaut sont celles de l'écran courant.

10.70.57 swf_ortho

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_ortho](#) (double *xmin* , double *xmax* , double *ymin* , double *ymax* , double

zmin , double ***zmax***)

[PHP 4 >= 4.0.1]

[swf_ortho\(\)](#) définit une projection orthogonale entre les coordonnées utilisateur et le port courant.

10.70.58 swf_ortho2

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_ortho2](#) (double ***xmin*** , double ***xmax*** , double ***ymin*** , double ***ymax***)

[PHP 4 >= 4.0RC2]

[swf_ortho2\(\)](#) définit une projection orthogonale à 2 dimensions entre les coordonnées utilisateur et le port courant. C'est la projection par défaut des animations Flash. Si vous souhaitez une perspective, utilisez plutôt [swf_perspective\(\)](#).

10.70.59 swf_perspective

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_perspective](#) (double ***fovy*** , double ***aspect*** , double ***near*** , double ***far***)

[PHP 4 >= 4.0RC2]

[swf_perspective\(\)](#) définit une projection orthogonale à 3 dimensions entre les coordonnées utilisateur et le port courant. Le paramètre ***fovy*** est l'angle de vue de la direction y. Le paramètre ***aspect*** doit être choisi pour correspondre au ratio de la vue utilisée. ***near*** est le plan adjacent proche ***far*** est le plan adjacent distant. Note : *Diverses distortions peuvent apparaître lors de ce genre de projection, car Flash ne dispose que d'une matrice à 2 dimensions. Certaines distortions font vraiment tâche d'encre.*

10.70.60 swf_polarview

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_polarview](#) (double ***dist*** , double ***azimuth*** , double ***incidence*** , double ***twist***)

[PHP 4 >= 4.0RC2]

[swf_polarview\(\)](#) définit la position de l'utilisateur en coordonnées polaires. ***dist*** est la distance entre le point de vue et l'origine. ***azimuth*** définit l'angle azimutal dans le plan x,y mesuré en distance depuis l'axe y. ***incidence*** définit l'angle d'incidence dans le plan y,z, mesuré en distance depuis l'axe z. Finalement, ***twist*** est l'angle de rotation du point de vue sur la ligne de vue, en utilisant la règle de la main droite.

10.70.61 swf_lookat

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_lookat](#) (double ***view_x*** , double ***view_y*** , double ***view_z*** , double ***reference_x*** , double ***reference_y*** , double ***reference_z*** , double ***twist***)

[PHP 4 >= 4.0RC2]

[swf_lookat\(\)](#) définit une transformation de vue, en donnant la position de la vue, de coordonnées (***view_x***, ***view_y*** et ***view_z***) et les coordonnées du point de référence dans la scène, de coordonnées (***reference_x***,

reference_y, *reference_z*). Le paramètre *twist* contrôle la rotation le long de l'axe des z de l'utilisateur.

10.70.62 swf_pushmatrix

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_pushmatrix](#)

[PHP 4 >= 4.0RC2]

[swf_pushmatrix\(\)](#) empile la matrice de transformation courante dans la pile.

10.70.63 swf_popmatrix

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_popmatrix](#)

[PHP 4 >= 4.0RC2]

[swf_popmatrix\(\)](#) dépile la matrice de transformation.

10.70.64 swf_scale

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_scale](#)(double *x* , double *y* , double *z*)

[PHP 4 >= 4.0RC2]

[swf_scale\(\)](#) fait une mise à l'échelle de *x* pour les coordonnées x, de *y* pour les coordonnées y et *z* pour les coordonnées z.

10.70.65 swf_translate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_translate](#)(double *x* , double *y* , double *z*)

[PHP 4 >= 4.0RC2]

[swf_translate\(\)](#) déplace la transformation courante de *x*, *y* et *z*, dans les directions x, y et z.

10.70.66 swf_rotate

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_rotate](#)(double *angle* , string *axis*)

[PHP 4 >= 4.0RC2]

[swf_rotate\(\)](#) fait subir la rotation d'angle *angle*, autour de l'axe *axis*. Les valeurs possibles pour *axis* sont : 'x' (axe x), 'y' (axe y) ou 'z' (axe z).

10.70.67 swf_posround

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [swf_posround](#)(int *round*)

[PHP 4 >= 4.0RC2]

[swf_posround\(\)](#) active ou désactive l'approximation lors des translations, lorsque des objets sont placés ou déplacés. Il y a des situations où le texte devient plus lisible lorsque l'approximation a été activée.

round active l'approximation (1) ou la désactive (0).

10.71 Sybase

[\[Notes en ligne\]](#)

10.71.1 sybase_affected_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_affected_rows](#)(int *link_identifieur*)

[PHP 3 >= 3.0.6, PHP 4]

[sybase_affected_rows\(\)](#) retourne le nombre de lignes affectées par la dernière requête.

[sybase_affected_rows\(\)](#) retourne le nombre de lignes affectées par la dernière requête INSERT, UPDATE ou DELETE sur le serveur associé à l'identifiant de connexion *link_identifieur*. Si le lien n'est pas précisé, le dernier lien ouvert est utilisé.

Cette commande ne sert à rien sur les requête SELECT : uniquement sur des requêtes qui modifient les lignes. Pour connaître le nombre de lignes retournées par un SELECT, utilisez [sybase_num_rows\(\)](#). Note : [sybase_affected_rows\(\)](#) est disponible avec l'interface CT vers Sybase, mais pas avec la librairie DB.

10.71.2 sybase_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_close](#)(int *link_identifieur*)

[PHP 3, PHP 4]

[sybase_close\(\)](#) retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[sybase_close\(\)](#) termine la connexion avec le serveur Sybase associé à l'identifiant de connexion *link_identifieur*.

Notez qu'il n'est pas utile de fermer les connexions non persistantes, car elles seront terminées à la fin du script.

[sybase_close\(\)](#) ne ferme pas les connexions persistantes générées par [sybase_pconnect\(\)](#).

Voir aussi : [sybase_connect\(\)](#) et [sybase_pconnect\(\)](#).

10.71.3 sybase_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_connect](#)(string *servername*, string *username*, string *password*)

[PHP 3, PHP 4]

[sybase_connect\(\)](#) retourne un identifiant positif de lien Sybase en cas de succès, et FALSE en cas d'erreur.
[sybase_connect\(\)](#) établit une connexion à un serveur Sybase. Le nom de serveur *servername* doit être valide, défini dans le fichier d'interface.

Si un deuxième appel à [sybase_connect\(\)](#) est fait avec les mêmes arguments, une nouvelle connexion ne sera pas établie, mais ce sera l'identifiant de la connexion déjà ouverte qui sera retourné.

La connexion sera fermée dès la fin du script, à moins qu'elle ne soit pas explicitement fermée avec [sybase_close\(\)](#).

Voir aussi [sybase_pconnect\(\)](#) et [sybase_close\(\)](#).

10.71.4 sybase_data_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_data_seek](#)(int *result_identifieur*, int *row_number*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[sybase_data_seek\(\)](#) déplace le pointeur interne de ligne du résultat Sybase associé à *result_identifieur* jusqu'à la ligne *row_number*. Le prochain appel à [sybase_fetch_row\(\)](#) sans préciser la ligne, retournera la ligne *row_number*.

Voir aussi: [sybase_data_seek\(\)](#).

10.71.5 sybase_fetch_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sybase_fetch_array](#)(int *result*)

[PHP 3, PHP 4]

Retourne un tableau qui contient la ligne demandée, ou FALSE s'il ne reste plus de ligne.

[sybase_fetch_array\(\)](#) est une version évoluée de [sybase_fetch_row\(\)](#). En plus d'enregistrer les données dans un tableau à index numérique, cette fonction peut aussi les enregistrer dans un tableau associatif, en utilisant les nom des champs comme clés.

Il est très important de noter que [sybase_fetch_array\(\)](#) N'est PAS nettement plus lent que [sybase_fetch_row\(\)](#), tandis qu'elle fournit un confort d'utilisation notable.

Pour plus de détails : [sybase_fetch_row\(\)](#).

10.71.6 sybase_fetch_field

[\[Notes en ligne\]](#) [\[Exemples\]](#)

object [sybase_fetch_field](#)(int *result*, int *field_offset*)

[PHP 3, PHP 4]

Retourne un objet contenant les informations du champs.

[sybase_fetch_field\(\)](#) sert à obtenir des informations à propos des champs dans le résultat *result*. Si l'offset du champs n'est pas précisé, le champs suivant est traité.

Les propriétés des objets sont :

- name – column name. nom de la colonne. Si la colonne est un résultat de fonction, le nom de cette fonction devient computed#N, où #N est un numéro de série.

- `column_source` – la table d'origine de la colonne.
- `max_length` – taille maximale de la colonne
- `numeric` – 1 si la colonne est de type numérique.
- `type` – type de données de la colonne

Voir aussi [sybase_field_seek\(\)](#).

10.71.7 sybase_fetch_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_fetch_object](#)(int *result*)

[PHP 3, PHP 4]

[sybase_fetch_object\(\)](#) retourne un objet qui contient la ligne demandée, en cas de succès, et FALSE en cas d'erreur.

[sybase_fetch_object\(\)](#) est similaire à [sybase_fetch_array\(\)](#), avec une différence : c'est un objet qui est retourné à la place d'un tableau. Indirectement, cela signifie que vous ne pourrez accéder aux valeurs que par les propriétés, et non plus avec des offsets (les nombres sont interdits comme nom de propriété).

Au niveau de la vitesse, cette fonction est identique à [sybase_fetch_array\(\)](#), et presque aussi rapide que [sybase_fetch_row\(\)](#) (la différence est insignifiante).

Voir aussi: [sybase_fetch_array\(\)](#) et [sybase_fetch_row\(\)](#).

10.71.8 sybase_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [sybase_fetch_row](#)(int *result*)

[PHP 3, PHP 4]

[sybase_fetch_row\(\)](#) retourne un tableau qui contient la ligne demandée, en cas de succès, et FALSE en cas d'erreur.

[sybase_fetch_row\(\)](#) lit une ligne dans le résultat associé à l'identifiant de résultat *result*. La ligne retournée est sous la forme d'un tableau. Chaque champs est enregistré dans un index du tableau, les index commençant à 0.

Les prochains appels à [sybase_fetch_row\(\)](#) retourneront la ligne suivante du résultat, ou FALSE, si il ne reste plus de lignes.

Voir aussi: [sybase_fetch_array\(\)](#), [sybase_fetch_object\(\)](#), [sybase_data_seek\(\)](#) et [sybase_result\(\)](#).

10.71.9 sybase_field_seek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_field_seek](#)(int *result*, int *field_offset*)

[PHP 3, PHP 4]

[sybase_field_seek\(\)](#) modifie l'index d'un champs. Le prochain appel à la fonction [sybase_fetch_field\(\)](#) sans préciser l'index du champs retournera ce champs.

Voir aussi: [sybase_fetch_field\(\)](#).

[10.71.10 sybase_free_result](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_free_result](#)(int *result*)

[PHP 3, PHP 4]

[sybase_free_result\(\)](#) n'est vraiment utile que si vous risquez d'utiliser trop de mémoire durant votre script. La mémoire occupée par les résultats est automatiquement libérée à la fin du script. Mais, si vous êtes sûr de ne pas avoir besoin du résultat ultérieurement.

[10.71.11 sybase_get_last_message](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sybase_get_last_message](#)(void)

[PHP 3, PHP 4]

[sybase_get_last_message\(\)](#) retourne le dernier message rapporté par le serveur.

[10.71.12 sybase_min_client_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_client_severity](#)(int *severity*)

[PHP 3, PHP 4]

[sybase_min_client_severity\(\)](#) fixe la sévérité minimale du client. Note : *Cette fonction est disponible avec l'interface CT vers Sybase, mais pas avec la librairie DB.*

See also: [sybase_min_server_severity\(\)](#).

[10.71.13 sybase_min_error_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_error_severity](#)(int *severity*)

[PHP 3, PHP 4]

[sybase_min_error_severity\(\)](#) fixe la sévérité minimale du client pour les erreurs.

Voir aussi: [sybase_min_message_severity\(\)](#).

[10.71.14 sybase_min_message_severity](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_message_severity](#)(int *severity*)

[PHP 3, PHP 4]

[sybase_min_message_severity\(\)](#) fixe la sévérité minimale du client pour les messages.

Voir aussi: [sybase_min_error_severity\(\)](#).

10.71.15 sybase_min_server_severity

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [sybase_min_server_severity](#) (int *severity*)

[PHP 3, PHP 4]

[sybase_min_server_severity\(\)](#) fixe la sévérité minimale du client pour le serveur. Note : *Cette fonction est disponible avec l'interface CT vers Sybase, mais pas avec la librairie DB.*

Voir aussi: [sybase_min_client_severity\(\)](#).

10.71.16 sybase_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_num_fields](#) (int *result*)

[PHP 3, PHP 4]

[sybase_num_fields\(\)](#) retourne le nombre de champs dans un résultat.

Voir aussi: [sybase_query\(\)](#), [sybase_fetch_field\(\)](#) et [sybase_num_rows\(\)](#).

10.71.17 sybase_num_rows

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_num_rows](#) (int *result*)

[PHP 3, PHP 4]

[sybase_num_rows\(\)](#) retourne le nombre de lignes dans un résultat.

Voir aussi: [sybase_query\(\)](#) et [sybase_fetch_row\(\)](#).

10.71.18 sybase_pconnect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_pconnect](#) (string *servername*, string *username*, string *password*)

[PHP 3, PHP 4]

Retourne un identifiant de connexion positive en cas de succès, et FALSE en cas d'erreur.

[sybase_pconnect\(\)](#) se comporte comme [sybase_connect\(\)](#) avec deux différences majeures :

Premièrement, lors de la connexion, la fonction va chercher une connexion (persistante) déjà ouverte, avec le même hôte, nom de compte et mot de passe. Si une telle connexion est trouvée, un identifiant de cette connexion est retourné, plutôt que d'en ouvrir une nouvelle.

Deuxièmement, la connexion au serveur SyBase ne sera pas terminée lors de la fin du script. Au contraire, le lien sera maintenu pour des connexions ultérieures. [sybase_close\(\)](#) ne fermera pas un lien créé par [sybase_pconnect\(\)](#).

Ce type de liens est dit 'persistant'.

10.71.19 sybase_query

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [sybase_query](#) (string *query*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne un identifiant de résultat positif en cas de succès, et FALSE sinon.

[sybase_query\(\)](#) envoie une requête à la base de données courante, sur le serveur associé à l'identifiant de connexion. Si l'identifiant de connexion n'est pas précisé, la fonction essaiera d'utiliser la dernière connexion ouverte. Si aucune connexion n'a été ouverte, la fonction va tenter d'ouvrir une connexion avec la fonction [sybase_connect\(\)](#).

Voir aussi: [sybase_select_db\(\)](#) et [sybase_connect\(\)](#).

10.71.20 sybase_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [sybase_result](#) (int *result*, int *i*, mixed *field*)

[PHP 3, PHP 4]

[sybase_result\(\)](#) retourne le contenu d'une cellule. L'argument *field* peut être l'index du champs, ou bien le nom du champs, ou encore, le nom de la table " point " le nom du champs. Si la colonne a été aliasée ('SELECT foo AS bar FROM...'), utilisez l'alias à la place du nom de la colonne.

Lorsque vous travaillez sur des résultats de grande taille, vous devriez utiliser les autres fonctions qui lisent une ligne entière (voir plus loin). Etant donné que ces fonctions lisent une ligne entière, elles sont BEAUCOUP plus rapide que [sybase_result\(\)](#). De plus, l'utilisation d'index numérique est beaucoup plus rapide que les noms des champs, ou les noms des tables et des champs.

Fonctions de substitution, à haute efficacité : [sybase_fetch_row\(\)](#), [sybase_fetch_array\(\)](#) et [sybase_fetch_object\(\)](#).

10.71.21 sybase_select_db

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [sybase_select_db](#) (string *database_name*, int *link_identifier*)

[PHP 3, PHP 4]

Retourne TRUE en cas de succès, et FALSE en cas d'erreur.

[sybase_select_db\(\)](#) change la base de données courante et active sur le serveur associé avec l'identifiant de connexion *link_identifier*. Si *link_identifier* n'est pas précisé, le dernier lien ouvert est utilisé. Si aucun lien n'a été ouvert, la fonction va tenter d'en établir un en appelant [sybase_connect\(\)](#).

Tous les prochains appels à [sybase_query\(\)](#) seront faites sur la bas de données courante et active.

Voir aussi : [sybase_connect\(\)](#), [sybase_pconnect\(\)](#) et [sybase_query\(\)](#).

10.72 ODBC unifié

[\[Notes en ligne\]](#)

En plus du support de l'ODBC normal, l'ODBC unifié de PHP vous donne accès à diverses bases de données qui ont emprunté la sémantique des API ODBC pour implémenter leur propres API. Au lieu de maintenir de

multiples pilotes qui sont similaires, ces pilotes ont été rassemblés dans un jeu de fonctions ODBC uniques. Les bases de données suivantes sont supportées par l'ODBC unifié : [Adabas D](#), [IBM DB2](#), [iODBC](#), [Solid](#), et [Sybase SQL Anywhere](#).

Reportez vous à [4.2.5 Options de base de données](#) pour plus de détails sur les configurations de ces serveurs.

Note : *Il n'y a pas d'ODBC utilisé lors des connexions aux bases de données ci-dessus. Les fonctions que vous utiliserez portent des noms évocateurs, et utilisent les mêmes syntaxes que leurs cousines d'ODBC.*

10.72.1 [odbc_autocommit](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_autocommit](#)(int *connection_id*, int *OnOff*)

[PHP 3>= 3.0.6, PHP 4]

Sans le paramètre *OnOff*, [odbc_autocommit\(\)](#) retourne le statut d'auto-validation de la connexion *connection_id*. TRUE si le mode est activé, FALSE si il ne l'est pas, ou si une erreur survient.

Si *OnOff* vaut TRUE, l'auto-validation est activée. Si il est FALSE, l'auto-validation est désactivée.

Retourne TRUE en cas de succès, FALSE en cas d'échec.

Par défaut, l'auto-validation est activée. Désactiver l'auto-validation est équivalent à démarrer une transaction.

Voir aussi [odbc_commit\(\)](#) et [odbc_rollback\(\)](#).

10.72.2 [odbc_binmode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_binmode](#)(int *result_id*, int *mode*)

[PHP 3>= 3.0.6, PHP 4]

Types ODBC SQL affectés: BINARY, VARBINARY, LONGVARBINARY.

- ODBC_BINMODE_PASSTHRU: Mode Passthru
- ODBC_BINMODE_RETURN: Retourne tel quel.
- ODBC_BINMODE_CONVERT: Convertit en char et retourne la valeur.

Lorsqu'une donnée SQL est convertie en caractère C, les 8 bits du caractère source sont représentés par deux caractères ASCII. Ces caractères sont des représentations ASCII des nombres au format hexadécimal. Par exemple, le binaire 00000001 est converti en "01" et le binaire 11111111 est converti en "FF".

mode	longueur	résultat
ODBC_BINMODE_PASSTHRU	0	passthru
ODBC_BINMODE_RETURN	0	passthru
ODBC_BINMODE_CONVERT	0	passthru
ODBC_BINMODE_PASSTHRU	0	passthru
ODBC_BINMODE_PASSTHRU	>0	passthru
ODBC_BINMODE_RETURN	>0	Tel quel
ODBC_BINMODE_CONVERT	>0	Caractère

Si [odbc_fetch_into\(\)](#) est utilisé, passthru signifie qu'une chaîne vide sera retournée pour ces colonnes.

Si **result_id** vaut 0, ces paramètres seront appliqués aux nouveaux résultats. Note : *La valeur par défaut de 4096 est 4096 et les valeurs par défaut de odbc_binmode est ODBC_BINMODE_RETURN. La gestion des colonnes binaires est aussi modifié par [odbc_longreadlen\(\)](#).*

10.72.3 odbc_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [odbc_close](#)(int **connection_id**)

[PHP 3>= 3.0.6, PHP 4]

[odbc_close\(\)](#) ferme la connexion avec une source de données, représentée par l'identifiant de connexion. Note : *[odbc_close\(\)](#) échouera si il y a des transactions en cours sur cette connexion. Dans ce cas, la connexion restera ouverte.*

10.72.4 odbc_close_all

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [odbc_close_all](#)

[PHP 3>= 3.0.6, PHP 4]

[odbc_close_all\(\)](#) ferme toutes les connexions ODBC à des sources de données. Note :

[odbc_close_all\(\)](#) échouera si il y a des transactions en cours sur cette connexion. Dans ce cas, la connexion restera ouverte.

10.72.5 odbc_commit

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_commit](#)(int **connection_id**)

[PHP 3>= 3.0.6, PHP 4]

Retourne TRUE en case de succès, FALSE en cas d'erreur. Toutes les connexions en cours sur **connection_id** sont validées.

10.72.6 odbc_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_connect](#)(string **dsn**, string **user**, string **password**, int **cursor_type**)

[PHP 3>= 3.0.6, PHP 4]

[odbc_connect\(\)](#) retourne un identifiant de connexion ODBC ou 0 (FALSE) en cas d'erreur.

L'identifiant de connexion retournée par cette fonction est nécessaire pour toutes les autres fonctions ODBC. Vous pouvez avoir de multiples connexions en même temps. Le quatrième paramètre fixe le type de pointeur de résultat utilisé pour cette connexion. Ce paramètre n'est généralement pas nécessaire, mais il peut être utile

pour contourner certains problèmes ODBC.

Avec certains pilotes ODBC, l'exécution de procédures enregistrées complexes peut produire l'erreur suivante : "Cannot open a cursor on a stored procedure that has anything other than a single select statement in it", ce qui signifie : "Impossible de créer un pointeur de résultat dans une procédure enregistrée qui est réduite à une simple sélection (SELECT)). Utiliser l'option `SQL_CUR_USE_ODBC` permet d'éviter cette erreur. De plus, certains pilotes ne supportent le paramètre optionnel de numéro de ligne dans [odbc_fetch_row\(\)](#).

`SQL_CUR_USE_ODBC` peut aussi permettre de résoudre ces problèmes.

Les constantes suivantes sont définies comme type de pointeur :

- `SQL_CUR_USE_IF_NEEDED`
- `SQL_CUR_USE_ODBC`
- `SQL_CUR_USE_DRIVER`
- `SQL_CUR_DEFAULT`

Pour les connexions persistantes, reportez vous à [odbc_pconnect\(\)](#).

[10.72.7 odbc_cursor](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_cursor](#) (int *result_id*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_cursor\(\)](#) lit le pointeur de fiche courante (cursorname) pour le résultat *result_id*.

[10.72.8 odbc_do](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_do](#) (int *conn_id*, string *query*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_do\(\)](#) exécute la requête *query* avec la connexion *conn_id*.

[10.72.9 odbc_error](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_error](#) (int *connection_id*)

[odbc_error\(\)](#) retourne un état ODBC sur 6 chiffres, ou une chaîne vide s'il n'y avait plus d'erreurs. Si *connection_id* est spécifié, le dernier état ODBC de cette connexion est retourné. Si *connection_id* est omis, c'est le dernier état de n'importe quelle connexion qui est retourné.

Voir aussi : [odbc_errormsg\(\)](#) et [odbc_exec\(\)](#).

[10.72.10 odbc_errormsg](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_errormsg](#) (int *connection_id*)

[odbc_errormsg\(\)](#) retourne une chaîne contenant le dernier message d'erreur ODBC, ou une chaîne vide s'il n'y avait pas d'erreur. Si *connection_id* est spécifié, le dernier état ODBC de cette connexion est retourné. Si *connection_id* est omis, c'est le dernier état de n'importe quelle connexion qui est retourné.

Voir aussi : [odbc_error\(\)](#) et [odbc_exec\(\)](#).

10.72.11 odbc_exec

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_exec](#) (int *connection_id*, string *query_string*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_exec\(\)](#) retourne FALSE en cas d'erreur, ou bien retourne un identifiant de résultat ODBC en cas d'exécution réussie.

[odbc_exec\(\)](#) envoie une commande SQL à la source de données représentée par *connection_id*. Ce paramètre doit être un identifiant valide de connexion, retourné par [odbc_connect\(\)](#) ou [odbc_pconnect\(\)](#).

Voir aussi : [odbc_prepare\(\)](#) et [odbc_execute\(\)](#) pour les exécutions multiples de requêtes SQL.

10.72.12 odbc_execute

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_execute](#) (int *result_id*, array *parameters_array*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_execute\(\)](#) exécute une requête SQL préparée par [odbc_prepare\(\)](#). Retourne TRUE en cas d'exécution réussie, et FALSE sinon. Le tableau de paramètres *parameters_array* ne sert que si vous avez besoin de paramètres votre requête.

10.72.13 odbc_fetch_into

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_fetch_into](#) (int *result_id*, int *rownumber*, array *result_array*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_fetch_into\(\)](#) retourne le nombre de colonnes dans le résultat, ou FALSE en cas d'erreur.

result_array doit avoir été passé par référence, mais il peut être de n'importe quel type, étant donné qu'il sera converti en tableau. Le tableau contiendra les valeurs des colonnes, ces dernières étant numérotées à partir de 0.

10.72.14 odbc_fetch_row

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_fetch_row](#) (int *result_id*, int *row_number*)

[PHP 3>= 3.0.6, PHP 4]

Si [odbc_fetch_row\(\)](#) a réussi, TRUE est retourné. Si il n'y avait plus de ligne, ou en cas d'erreur, FALSE est retourné.

[odbc_fetch_row\(\)](#) lit une ligne dans le résultat identifié par *result_id* et retourné par [odbc_do\(\)](#) ou [odbc_exec\(\)](#). Après [odbc_fetch_row\(\)](#), les champs seront accessibles avec la fonction [odbc_result\(\)](#).

Si *row_number* est omis, *row_number* va tenter de lire la prochaine ligne dans le résultat. Des appels répétés à [odbc_fetch_row\(\)](#) avec et sans paramètre *row_number* peuvent être combinés librement.

Pour passer en revue toutes les lignes d'un résultat plusieurs fois, vous pouvez appeler [odbc_fetch_row\(\)](#) avec *row_number* = 1, puis continue à appeler [odbc_fetch_row\(\)](#) sans le paramètre *row_number* pour passer en revue tout le résultat. Si un pilote ne supporte pas la lecture des lignes par numéro, le paramètre sera ignoré.

[10.72.15 odbc_field_name](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_field_name](#)(int *result_id*, int *field_number*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_field_name\(\)](#) lit le nom de la colonne dont l'index est *field_number*. La numérotation des champs commence à 1. FALSE est retourné en cas d'erreur.

[10.72.16 odbc_field_num](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_field_num](#)(int *result_id*, string *field_name*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_field_num\(\)](#) retourne le numéro de la colonne nommée *field_name*. Ce numéro correspond à l'index du champs dans le résultat ODBC. La numérotation commence à 1. FALSE est retourné en cas d'erreur.

[10.72.17 odbc_field_type](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_field_type](#)(int *result_id*, int *field_number*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_field_type\(\)](#) retourne le type de données SQL d'un champs, identifié par son index. La numérotation des champs commence à 1.

[10.72.18 odbc_field_len](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_field_len](#)(int *result_id*, int *field_number*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_field_len\(\)](#) retourne la longueur du champs référence par le nombre *field_number*, dans la connexion ODBC *result_id*. Les numéros de champs commencent à 1.

[10.72.19 odbc_field_precision](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_field_precision](#)(int *result_id*, int *field_number*)

[PHP 4 >= 4.0.0]

[odbc_field_precision\(\)](#) retourne la précision du champs référencé par son numéro *field_number*, dans le résultat ODBC *result_id*.

Voir aussi : [odbc_field_scale\(\)](#) pour connaître l'échelle d'un nombre à virgule flottante.

10.72.20 odbc_field_scale

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_field_scale](#) (int *result_id*, int *field_number*)

[PHP 4 >= 4.0.0]

[odbc_field_precision\(\)](#) retourne l'échelle du champs référencé par son numéro de champs *field_number* dans le résultat ODBC *result_id*.

10.72.21 odbc_free_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_free_result](#) (int *result_id*)

[PHP 3 >= 3.0.6, PHP 4]

Retourne toujours TRUE.

[odbc_free_result\(\)](#) n'est nécessaire que si vous craignez d'utiliser trop de mémoire lors de l'exécution de votre script. Tous les résultats en mémoire seront libérés dès la fin du script. Mais, si vous êtes sûr que vous n'aurez plus besoin d'un résultat jusqu'à la fin de votre script, vous pouvez appeler [odbc_free_result\(\)](#), et la mémoire associée à *result_id* sera libérée.

Note : Si *auto-validation* est désactivée (voir [odbc_autocommit\(\)](#)) et que vous appelez [odbc_free_result\(\)](#) avant de valider vos requêtes, toutes les transactions préparées seront annulées.

10.72.22 odbc_longreadlen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_longreadlen](#) (int *result_id*, int *length*)

[PHP 3 >= 3.0.6, PHP 4]

Types ODBC SQL affectés: LONG, LONGVARBINARY.

Le nombre d'octets retournés à PHP est contrôlé par le paramètre *length*. Si sa valeur est 0, les colonnes de type Long seront transformées en chaîne vide.

Note : La gestion des types LONGVARBINARY est aussi affectée par [odbc_binmode\(\)](#).

10.72.23 odbc_num_fields

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_num_fields](#) (int *result_id*)

[PHP 3 >= 3.0.6, PHP 4]

[odbc_num_fields\(\)](#) retourne le nombre de colonnes dans un résultat ODBC. Cette fonction retournera -1 en cas d'erreur. L'argument est un identifiant de résultat valide, retourné par [odbc_exec\(\)](#).

[10.72.24 `odbc_pconnect`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_pconnect](#)(string *dsn*, string *user*, string *password*, int *cursor_type*)

[PHP 3>= 3.0.6, PHP 4]

Retourne un identifiant de connexion ODBC ou 0 (FALSE) en cas d'erreur. Cette fonction se comporte de manière similaire à [odbc_connect\(\)](#), mais la connexion ouverte n'est pas vraiment terminée lorsque le script est terminé. Les prochaines requêtes qui se feront sur une connexion dont les *dsn*, *user*, *password* sont les mêmes que celle-ci (avec [odbc_connect\(\)](#) et [odbc_pconnect\(\)](#)) réutiliseront la connexion ouverte.

Note : *Les connexions persistantes n'ont aucun effet si PHP est utilisé comme CGI.*

Pour plus de détails sur le paramètre optionel *cursor_type*, voyez [odbc_connect\(\)](#). Pour plus de détails sur les connexions persistantes, reportez vous à la FAQ PHP.

[10.72.25 `odbc_prepare`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_prepare](#)(int *connection_id*, string *query_string*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_prepare\(\)](#) prépare une commande pour l'exécution.

Retourne un identifiant de résultat ODBC si la commande SQL a été préparée avec succès. L'identifiant peut être utilisé plus tard pour exécuter la commande avec [odbc_execute\(\)](#).

[10.72.26 `odbc_num_rows`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_num_rows](#)(int *result_id*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_num_rows\(\)](#) retourne le nombre de lignes dans un résultat ODBC. Cette fonction retournera -1 en cas d'erreur. Pour les commandes INSERT, UPDATE et DELETE, [odbc_num_rows\(\)](#) retourne le nombre de ligne affectées. Pour les commandes SELECT, ce PEUT le nombre de lignes disponibles, mais ce n'est pas certain.

Note: [odbc_num_rows\(\)](#) après un SELECT retournera -1 avec de nombreux pilotes.

[10.72.27 `odbc_result`](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [odbc_result](#)(int *result_id*, mixed *field*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_result\(\)](#) retourne le contenu d'un champs.

field peut être aussi bien un entier, contenant le numéro de colonne du champs, dans le résultat, ou bien une chaîne de caractère, qui représente le nom du champs. Par exemple:

```
<?php
```

```
$item_3 = odbc_result($Query_ID, 3 );
$item_val = odbc_result($Query_ID, "val");
?>
```

Le premier appel à [odbc_result\(\)](#) retourne la valeur du troisième champs de la ligne courante, du résultat *result_id*. Le deuxième appel à [odbc_result\(\)](#) retourne la valeur du troisième champs dont le nom est "val" de la ligne courante, du résultat *result_id*. Une erreur survient si le paramètre de colonne est inférieur à 1, ou dépasse le nombre de colonnes du résultat. De la même manière, une erreur survient si le nom du champs passé ne correspond à aucun champs dans le résultat.

Les index de champs commencent à 1. Pour plus d'informations sur la façon de lire des colonnes de type binaire ou long, reportez vous à [odbc_binmode\(\)](#) et [odbc_longreadlen\(\)](#).

[10.72.28 odbc_result_all](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_result_all](#) (int *result_id*, string *format*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_result_all\(\)](#) retourne le nombre de lignes dans le résultat, ou FALSE en cas d'erreur.

[odbc_result_all\(\)](#) affiche toutes les lignes d'un résultat. L'affichage se fait au format HTML. Avec l'option *format*, il est possible de modifier l'aspect global de la table.

[10.72.29 odbc_rollback](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_rollback](#) (int *connection_id*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_rollback\(\)](#) annule toutes les transactions sur la connexion *connection_id*. Retourne TRUE en cas de succès, et FALSE en cas d'échec.

[10.72.30 odbc_setoption](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_setoption](#) (int *id*, int *function*, int *option*, int *param*)

[PHP 3>= 3.0.6, PHP 4]

[odbc_setoption\(\)](#) donne accès aux options ODBC pour une connexion particulière ou un résultat de requête.

Elle a été écrite pour aider à la résolution de problème liés aux pilotes ODBC récalcitrants. Vous aurez sûrement à utiliser cette fonction si vous êtes un programmeur ODBC et que vous comprenez les divers effets des options disponibles. Vous aurez aussi besoin d'un bon manuel de référence pour comprendre les options et leur usage. Différentes versions de pilotes supportent différentes versions d'options.

Etant donné que les effets peuvent varier d'un pilote à l'autre, l'utilisation de cette fonction dans des scripts voués à être livrés au public est très fortement déconseillée. De plus, certaines options ODBC ne sont pas disponibles car elles doivent être fixées avant l'établissement de la connexion. Cependant, si dans un cas bien spécifique, cette fonction vous permet d'utiliser PHP sans que votre patron vous pousse à utiliser un produit commercial, alors cela n'a pas d'importance.

Id est un identifiant de connexion, ou un identifiant de résultat, pour lequel vous souhaitez modifier des

options. Pour `SQLSetConnectOption()`, c'est un identifiant de connexion. Pour `SQLSetStmtOption()`, c'est un identifiant de résultat.

function est la fonction ODBC à utiliser. La valeur doit être de 1 pour utiliser `SQLSetConnectOption()` et 2 pour `SQLSetStmtOption()`.

Le paramètre **option** est l'option à modifier.

Le paramètre **param** est la valeur de l'option **option**.

Exemple de modification d'option ODBC

```
<?php
// 1. L'option 102 de SQLSetConnectOption() est SQL_AUTOCOMMIT.
// 1 de SQL_AUTOCOMMIT est SQL_AUTOCOMMIT_ON.
// Cet exemple a le meme effet que
//      odbc_autocommit($conn, TRUE);
odbc_setoption($conn, 1, 102, 1);
// 2. Option 0 de SQLSetStmtOption() est SQL_QUERY_TIMEOUT.
// Cet exemple fixe le délai d'expiration à 30 secondes.
$result = odbc_prepare($conn, $sql);
odbc_setoption($result, 2, 0, 30);
odbc_execute($result);
?>
```

10.72.31 odbc_tables

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_tables](#) (int *connection_id*, string *qualifier*, string *owner*, string *name*, string *types*)

[PHP 3>= 3.0.17, PHP 4 >= 4.0b4]

[odbc_tables\(\)](#) liste toutes les tables de la source et retourne un identifiant de résultat ODBC, ou bien FALSE en cas d'erreur.

Le résultat contient les colonnes suivantes :

- TABLE_QUALIFIER
- TABLE_OWNER
- TABLE_NAME
- TABLE_TYPE
- REMARKS

Le résultat est ordonné grâce aux options TABLE_TYPE, TABLE_QUALIFIER, TABLE_OWNER et TABLE_NAME.

Les paramètres **owner** et **name** acceptent des masques de recherche ('%' pour remplacer zéro ou plus caractères, et '_' pour n'en remplacer qu'un seul).

Pour supporter les énumérations de qualifieurs, propriétaire et types de tables, la sémantique suivante pour les paramètres **qualifier**, **owner**, **name**, et **table_type** sont disponibles :

- Si **qualifier** est un signe de pourcentage (%), et **owner** et **name** sont des chaînes vides, alors le résultat contient la liste des qualifieurs valides pour la source. (toutes les colonnes hormis TABLE_QUALIFIER contiennent NULL).
- Si **owner** est un signe de pourcentage (%), et **qualifier** et **name** sont des chaînes vides, alors le

résultat contient la liste des propriétaires de la source (toutes les colonnes hormis TABLE_OWNER contiennent NULL).

- Si *table_type* est un signe de pourcentage (%), et *qualifier*, *owner* et *name* sont des chaînes vides, alors le résultat contient la liste des types de tables de la source (toutes les colonnes hormis TABLE_TYPE contiennent NULL).

Si *table_type* n'est pas une chaîne vide, il doit contenir une liste de valeurs, séparées par des virgules, qui représentent les types recherchés. Chaque valeur peut être insérée entre guillemets simples ('), ou sans guillemets. Par exemple "'TABLE','VIEW'" ou "TABLE, VIEW". Si la source de données ne supporte par un type de table donné, [odbc_tables\(\)](#) ne retournera aucun résultat pour ce type.

Voir aussi [odbc_tableprivileges\(\)](#) pour connaître les droits associés.

[10.72.32 odbc_tableprivileges](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_tableprivileges](#)(int *connection_id*, string *qualifier*, string *owner*, string *name*)

[PHP 4 >= 4.0b4]

[odbc_tableprivileges\(\)](#) liste les tables de la source et leurs droits associés. Retourne un identifiant de résultat ODBC, ou bien FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- TABLE_QUALIFIER
- TABLE_OWNER
- TABLE_NAME
- GRANTOR
- GRANTEE
- PRIVILEGE
- IS_GRANTABLE

Le résultat est ordonné par TABLE_QUALIFIER, TABLE_OWNER et TABLE_NAME.

Les paramètres *owner* et *name* acceptent des masques de recherche ('%' pour remplacer zéro ou plus caractères, et '_' pour n'en remplacer qu'un seul).

[10.72.33 odbc_columns](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_columns](#)(int *connection_id*, string *qualifier*, string *owner*, string *table_name*, string *column_name*)

[PHP 4 >= 4.0b4]

[odbc_columns\(\)](#) liste toutes les colonnes de la source de données. Retourne un identifiant de résultat ODBC, ou bien FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- TABLE_QUALIFIER

- TABLE_OWNER
- TABLE_NAME
- COLUMN_NAME
- DATA_TYPE
- TYPE_NAME
- PRECISION
- LENGTH
- SCALE
- RADIX
- NULLABLE
- REMARKS

Le résultat est ordonné par TABLE_QUALIFIER, TABLE_OWNER et TABLE_NAME.

Les paramètres *owner*, *column_name* et *table_name* acceptent des masques de recherche ('%' pour remplacer zéro ou plus caractères, et '_' pour n'en remplacer qu'un seul).

Voir aussi [odbc_columnprivileges\(\)](#) pour connaître les droits associés.

10.72.34 odbc_columnprivileges

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_columnprivileges](#) (int *connection_id*, string *qualifier*, string *owner*, string *table_name*, string *column_name*)

[PHP 4 >= 4.0b4]

[odbc_columnprivileges\(\)](#) liste les colonnes et leurs droits associés pour la table *table_name*. Retourne un identifiant de résultat ODBC, ou bien FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- TABLE_QUALIFIER
- TABLE_OWNER
- TABLE_NAME
- GRANTOR
- GRANTEE
- PRIVILEGE
- IS_GRANTABLE

Le résultat est ordonné par TABLE_QUALIFIER, TABLE_OWNER et TABLE_NAME.

Le paramètre *column_name* accepte des masques de recherche ('%' pour remplacer zéro ou plus caractères, et '_' pour n'en remplacer qu'un seul).

10.72.35 odbc_gettypeinfo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_gettypeinfo](#) (int *connection_id*, int *data_type*)

[PHP 4 >= 4.0b4]

[odbc_gettypeinfo\(\)](#) liste les types de données qui sont supportées par une source. Retourne un identifiant de

résultat, ou FALSE en cas d'erreur. L'argument optionnel *data_type* peut être utilisé pour restreindre les informations à un seul type de données.

Le résultat possède les colonnes suivantes :

- TYPE_NAME
- DATA_TYPE
- PRECISION
- LITERAL_PREFIX
- LITERAL_SUFFIX
- CREATE_PARAMS
- NULLABLE
- CASE_SENSITIVE
- SEARCHABLE
- UNSIGNED_ATTRIBUTE
- MONEY
- AUTO_INCREMENT
- LOCAL_TYPE_NAME
- MINIMUM_SCALE
- MAXIMUM_SCALE

Le résultat est ordonné par DATA_TYPE et TYPE_NAME.

[10.72.36 odbc_primarykeys](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_primarykeys](#) (int *connection_id*, string *qualifier*, string *owner*, string *table*)
[PHP 4 >= 4.0b4]

[odbc_primarykeys\(\)](#) liste les colonnes utilisées dans une clé primaire de la table *table*. Retourne un identifiant de résultat, ou FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- TABLE_QUALIFIER
- TABLE_OWNER
- TABLE_NAME
- COLUMN_NAME
- KEY_SEQ
- PK_NAME

[10.72.37 odbc_foreignkeys](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_foreignkeys](#) (int *connection_id*, string *pk_qualifier*, string *pk_owner*, string *pk_table*, string *fk_qualifier*, string *fk_owner*, string *fk_table*)
[PHP 4 >= 4.0b4]

[odbc_foreignkeys\(\)](#) liste les clés étrangères utilisées dans la table *pk_table*. Retourne un identifiant de résultat, ou FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- PKTABLE_QUALIFIER
- PKTABLE_OWNER
- PKTABLE_NAME
- PKCOLUMN_NAME
- FKTABLE_QUALIFIER
- FKTABLE_OWNER
- FKTABLE_NAME
- FKCOLUMN_NAME
- KEY_SEQ
- UPDATE_RULE
- DELETE_RULE
- FK_NAME
- PK_NAME

Si *pk_table* contient un nom de table, [odbc_foreignkeys\(\)](#) retourne la clé primaire de la table *pk_table*, et toutes les clés étrangères qui y font référence.

Si *fk_table* contient un nom de table, [odbc_foreignkeys\(\)](#) retourne la liste des clés étrangères de la table *fk_table*, et les clés primaires (d'autres tables) qui y font référence.

Si *pk_table* et *fk_table* contiennent des noms de tables, [odbc_foreignkeys\(\)](#) retourne la liste des clés étrangères de la table *fk_table* qui utilisent la clé primaire de la table *pk_table*. Cette liste devrait ne contenir qu'une clé au mieux.

10.72.38 odbc_procedures

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_procedures](#) (int *connection_id*, string *qualifier*, string *owner*, string *name*)
[PHP 4 >= 4.0b4]

[odbc_procedures\(\)](#) liste toutes les procédures stockées dans la source de données. Retourne un identifiant de résultat, ou FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- PROCEDURE_QUALIFIER
- PROCEDURE_OWNER
- PROCEDURE_NAME
- NUM_INPUT_PARAMS
- NUM_OUTPUT_PARAMS
- NUM_RESULT_SETS
- REMARKS
- PROCEDURE_TYPE

Les paramètres *owner* et *name* acceptent des masques de recherche ('%' pour remplacer zéro ou plus caractères, et '_' pour n'en remplacer qu'un seul).

10.72.39 odbc_procedurecolumns

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_procedurecolumns](#)(int *connection_id*, string *qualifier*, string *owner*, string *proc*, string *column*)

[PHP 4 >= 4.0b4]

[odbc_procedurecolumns\(\)](#) list les paramètres d'entrée et de sortie, ainsi que les colonnes utilisées dans les procédures désignées par les paramètres. Retourne un identifiant de résultat, ou FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- PROCEDURE_QUALIFIER
- PROCEDURE_OWNER
- PROCEDURE_NAME
- COLUMN_NAME
- COLUMN_TYPE
- DATA_TYPE
- TYPE_NAME
- PRECISION
- LENGTH
- SCALE
- RADIX
- NULLABLE
- REMARKS

Le résultat est ordonné par PROCEDURE_QUALIFIER, PROCEDURE_OWNER, PROCEDURE_NAME et COLUMN_TYPE.

Les paramètres *owner*, *proc* et *column* acceptent des masques de recherche ('%' pour remplacer zéro ou plus caractères, et '_' pour n'en remplacer qu'un seul).

10.72.40 odbc_specialcolumns

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_specialcolumns](#)(int *connection_id*, int *type*, string *qualifier*, string *owner*, string *table*, int *scope*, int *nullable*)

[PHP 4 >= 4.0b4]

Lorsque le *type* est SQL_BEST_ROWID, [odbc_specialcolumns\(\)](#) retourne la ou les colonnes qui permettent de repérer uniquement chaque ligne d'une table.

Lorsque le type *type* est SQL_ROWVER, [odbc_specialcolumns\(\)](#) retourne l'ensemble optimal de colonne tel qu'en lisant les valeurs de ces colonnes, on puisse spécifier n'importe quelle ligne de manière unique.

Retourne un identifiant de résultat, ou FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- SCOPE
- COLUMN_NAME
- DATA_TYPE
- TYPE_NAME

- PRECISION
- LENGTH
- SCALE
- PSEUDO_COLUMN

Le résultat est ordonné par SCOPE.

10.72.41 odbc_statistics

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [odbc_statistics](#) (int *connection_id*, string *qualifier*, string *owner*, string *table_name*, int *unique*, int *accuracy*)

[PHP 4 >= 4.0b4]

[odbc_statistics\(\)](#) effectue quelques statistiques sur une tables et ses index. Retourne un identifiant de résultat, ou FALSE en cas d'erreur.

Le résultat possède les colonnes suivantes :

- TABLE_QUALIFIER
- TABLE_OWNER
- TABLE_NAME
- NON_UNIQUE
- INDEX_QUALIFIER
- INDEX_NAME
- TYPE
- SEQ_IN_INDEX
- COLUMN_NAME
- COLLATION
- CARDINALITY
- PAGES
- FILTER_CONDITION

Le résultat est ordonné par NON_UNIQUE, TYPE, INDEX_QUALIFIER, INDEX_NAME et SEQ_IN_INDEX.

10.73 URL

[\[Notes en ligne\]](#)

10.73.1 base64_decode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [base64_decode](#) (string *encoded_data*)

[PHP 3, PHP 4]

[base64_decode\(\)](#) décode *encoded_data* et retourne les données décodées. Les informations initiales peuvent être binaires.

Voir aussi : [base64_encode\(\)](#) et la RFC–2045 section 6.8.

10.73.2 base64_encode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [base64_encode](#)(string *data*)
[PHP 3, PHP 4]

[base64_encode\(\)](#) retourne *data* encodé en base64. Cet encodage est fait pour permettre aux informations binaires d'être manipulées par les systèmes qui ne gèrent pas correctement les 8 bits, comme par exemple, les corps de mail.

Une chaîne encodée Base64 prend, grosso modo, 33% de plus que les données initiales.

Voir aussi: [base64_decode\(\)](#), [chunk_split\(\)](#), et la RFC–2045 section 6.8.

10.73.3 parse_url

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [parse_url](#)(string *url*)
[PHP 3, PHP 4]

[parse_url\(\)](#) retourne un tableau associatif contenant les composants de l'URL. Les composants recherchés sont : "scheme", "host", "port", "user", "pass", "path", "query", et "fragment".

10.73.4 rawurldecode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [rawurldecode](#)(string *str*)
[PHP 3, PHP 4]

Retourne une chaîne dont les séquences de caractères %xy, avec xy deux valeurs hexadécimales, auront été remplacées par le caractère ASCII correspondant. Par exemple, la chaîne

```
foo%20bar%40baz
```

devient @example foo bar@baz .

Voir aussi [rawurlencode\(\)](#), [urldecode\(\)](#), [urlencode\(\)](#).

10.73.5 rawurlencode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [rawurlencode](#)(string *str*)
[PHP 3, PHP 4]

Retourne une chaîne dont tous les caractères non–alpha–numériques (hormis @example –_ .) auront été remplacés par des séquences %xy (%), avec xy deux valeurs hexadécimales. Ce codage est conforme à la RFC1738 qui évite que les caractères spéciaux soient interprétés comme des délimiteurs, et pour protéger les URL lors du transfert (contrairement à certains systèmes email). Par exemple, si vous voulez mettre un mot de passe dans une URL de ftp :

Exemple avec rawurlencode()

```
echo '<A HREF="ftp://user:', rawurlencode ('foo @+%/'),
      '@ftp.my.com/x.txt">';
```

Ou, si vous transmettez un chemin dans une URL

Exemple avec rawurlencode()

```
echo '<A HREF="http://x.com/department_list_script/',
      rawurlencode ('sales et marketing/Miami'), '">';
```

Voir aussi [rawurldecode\(\)](#), [urldecode\(\)](#) et [urlencode\(\)](#).

10.73.6 urldecode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [urldecode](#) (string *str*)

[PHP 3, PHP 4]

Décode toutes les séquences %## et les remplace par leur valeur. La chaîne ainsi décodée est retournée.

Exemple avec urldecode()

```
<?php
$a = split ('', $querystring);
$i = 0;
while ($i < count ($a)) {
    $b = split ('=', $a [$i]);
    echo 'Value for parameter ', htmlspecialchars (urldecode ($b [0])),
          ' is ', htmlspecialchars (urldecode ($b [1])), "<BR>";
    $i++;
}
?>
```

Voir aussi [urlencode\(\)](#), [rawurlencode\(\)](#) et [rawurldecode\(\)](#).

10.73.7 urlencode

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [urlencode](#) (string *str*)

[PHP 3, PHP 4]

Retourne une chaîne dont les caractères non-alphanumériques (hormis -_.) sont remplacés par des séquences commençant par un caractère pourcentage (%), suivi de deux chiffres hexadécimaux. Les espaces sont remplacés par des signes plus (+). Ce codage est celui qui est utilisé pour poster des informations dans les formulaires HTML. Le type MIME est application/x-www-form-urlencoded. Ce codage est différent de celui spécifié dans la RFC1738 (voir [rawurlencode\(\)](#)) : pour des raisons historiques, les espaces sont remplacés par des signes plus (+). [urlencode\(\)](#) est pratique pour transmettre des informations via une URL. C'est aussi un moyen de passer des informations d'une page à l'autre.

Exemple avec urlencode()

```
echo '<A HREF="moncgi?foo=', urlencode ($userinput), '">';
```

Voir aussi [urldecode\(\)](#).

Note: Faites bien attention aux variables qui ressemblent à des entités HTML, comme par exemple &, © et £, qui sont analysées par le client web et remplacée par leur valeur, au lieu de passer le nom de variable désiré. C'est un vrai problème qui a été montré par le W3C depuis longtemps. La référence est ici : <http://www.w3.org/TR/html4/appendix/notes.html#h-B.2.2>. PHP supporte le remplacement de séparateur d'arguments par un point-virgule, comme recommandé par le W3C, grâce à la directive `arg_separator.ini`. Malheureusement, la plus part des clients web n'envoie pas leur données de formulaire avec des point-virgule. Une solution plus portable est d'utiliser & à la place de & comme séparateur. Vous n'avez alors pas à changer la directive `arg_separator`. Laissez la à &, mais encodez vos URL avec [htmlentities\(\)](#)(urlencode(\$data)).

Exemple avec `urlencode()` et `htmlentities()`

```
<?php
echo '<A HREF="moncgi?foo=', htmlentities (urlencode ($userinput) ), '">';
?>
```

Voir aussi [urldecode\(\)](#), [htmlentities\(\)](#), [rawurldecode\(\)](#) et [rawurlencode\(\)](#).

10.74 Variables

[\[Notes en ligne\]](#)

10.74.1 doubleval

[\[Notes en ligne\]](#) [\[Exemples\]](#)

double [doubleval](#) (mixed *var*)

[PHP 3, PHP 4]

[doubleval\(\)](#) retourne la valeur numérique (double) de la variable *var*.

var peut être de type scalaire. Vous ne pouvez pas utiliser la fonction [doubleval\(\)](#) avec un tableau ou un objet.

```
<?php
$var = '122.343431e';
$double_valeur_de_var = doubleval($var);
print $double_valeur_de_var; // affiche 122.34343
?>
```

Voir aussi [intval\(\)](#), [strval\(\)](#), [settype\(\)](#) et [9.8.6 Définition du type](#).

10.74.2 empty

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [empty](#) (mixed *var*)

[empty\(\)](#) retourne la valeur FALSE si la variable *var* est affectée ou bien a une valeur différente de 0; la valeur TRUE dans les autres cas.

```
<?php
$var = 0;
if (empty($var)) { // retourne TRUE
print 'soit $var vaut 0, soit il n'est pas défini';
}
if (!isset($var)) { // retourne FALSE
print '$var n'est pas définie';
}
?>
```

Notez que cette fonction n'a pas de sens si elle est utilisée sur autre chose qu'une variable. i.e. `empty(addslashes($name))` n'a pas de sens, car cela revient à vérifier une entité qui n'est pas une variable. Voir aussi [isset\(\)](#) et [unset\(\)](#).

10.74.3 gettype

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gettype](#) (mixed *var*)
[PHP 3, PHP 4]

Retourne le type de la variable PHP *var*.
Les chaînes de caractères que peut retourner la fonction sont les suivantes :

- "boolean"
- "integer"
- "double"
- "string"
- "array"
- "object"
- "resource"
- "user function"
- "unknown type"

Voir aussi [settype\(\)](#).

10.74.4 intval

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [intval](#) (mixed *var*, int *base*)
[PHP 3, PHP 4]

Retourne la valeur numérique (integer) de la variable *var*, en convertissant la valeur dans la base spécifiée (par défaut en base 10).

var peut être de type scalaire. Vous ne pouvez pas utiliser la fonction [intval\(\)](#) avec un tableau ou un objet. Voir aussi [doubleval\(\)](#), [strval\(\)](#), [settype\(\)](#) et [9.8.6 Définition du type](#).

10.74.5 is_array

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool is_array (mixed *var*)
[PHP 3, PHP 4]

Renvoie la valeur TRUE si la variable *var* est un tableau, FALSE sinon.

Voir aussi [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_long\(\)](#), et [is_object\(\)](#).

10.74.6 is_bool

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool is_bool (mixed *var*)
[PHP 4 >= 4.0b4]

[is_bool\(\)](#) retourne TRUE si *var* est un booléen.

Voir aussi [is_array\(\)](#), [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_long\(\)](#), et [is_object\(\)](#).

10.74.7 is_double

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool is_double (mixed *var*)
[PHP 3, PHP 4]

Renvoie TRUE si la variable *var* est du type "double", FALSE sinon.

Voir aussi [is_array\(\)](#), [is_bool\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_long\(\)](#), et [is_object\(\)](#).

10.74.8 is_float

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool is_float (mixed *var*)
[PHP 3, PHP 4]

Cette fonction est un alias de la fonction [is_double\(\)](#).

Voir aussi [is_double\(\)](#), [is_bool\(\)](#), [is_real\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_string\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_long\(\)](#).

10.74.9 is_int

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool is_int (mixed *var*)
[PHP 3, PHP 4]

[is_int\(\)](#) est un alias de la fonction [is_long\(\)](#).

Voir aussi [is_bool\(\)](#), [is_double\(\)](#), [is_float\(\)](#), [is_integer\(\)](#), [is_string\(\)](#), [is_real\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_long\(\)](#).

10.74.10 is_integer

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_integer](#)(mixed *var*)

[PHP 3, PHP 4]

Cette fonction est un alias de la fonction [is_long\(\)](#).

Voir aussi [is_bool\(\)](#), [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_string\(\)](#), [is_real\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_long\(\)](#).

10.74.11 is_long

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_long](#)(mixed *var*)

[PHP 3, PHP 4]

Renvoie TRUE si la variable *var* est du type integer (long), FALSE sinon.

Voir aussi [is_bool\(\)](#), [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_integer\(\)](#).

10.74.12 is_numeric

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_numeric](#)(mixed *var*)

[PHP 4 >= 4.0RC1]

[is_numeric\(\)](#) retourne TRUE si *var* est un nombre, ou une chaîne numérique, ou FALSE sinon.

Voir aussi [is_bool\(\)](#), [is_double\(\)](#), [is_float\(\)](#), [is_int\(\)](#), [is_real\(\)](#), [is_string\(\)](#), [is_object\(\)](#), [is_array\(\)](#), et [is_integer\(\)](#).

10.74.13 is_object

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_object](#)(mixed *var*)

[PHP 3, PHP 4]

Renvoie TRUE si la variable *var* est un objet, FALSE sinon.

Voir aussi [is_bool\(\)](#), [is_long\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_float\(\)](#), [is_double\(\)](#), [is_real\(\)](#), [is_string\(\)](#), et [is_array\(\)](#).

10.74.14 is_real

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_real](#)(mixed *var*)

[PHP 3, PHP 4]

Cette fonction est un alias de la fonction [is_double\(\)](#).

Voir aussi [is_bool\(\)](#), [is_long\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_float\(\)](#), [is_double\(\)](#), [is_object\(\)](#), [is_string\(\)](#), et [is_array\(\)](#).

[10.74.15 is_resource](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_resource](#) (mixed *var*)

[PHP 4 >= 4.0b4]

[is_resource\(\)](#) retourne TRUE si la variable *var* est une ressource PHP, sinon FALSE.

Les ressources peuvent être des pointeurs de fichiers, des identifiants de résultats SQL, qui sont allouées et libérées en interne, par PHP, et qui peuvent demander un peu de nettoyage lorsqu'elles sont devenues inutiles, mais pas encore supprimées.

[10.74.16 is_string](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [is_string](#) (mixed *var*)

[PHP 3, PHP 4]

Renvoie TRUE si la variable *var* est du type "string", FALSE sinon.

Voir aussi [is_long\(\)](#), [is_int\(\)](#), [is_integer\(\)](#), [is_float\(\)](#), [is_double\(\)](#), [is_real\(\)](#), [is_object\(\)](#), et [is_array\(\)](#).

[10.74.17 isset](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [isset](#) (mixed *var*)

Renvoie TRUE si la variable *var* est définie, FALSE sinon.

Si une variable a été désaffectée avec la fonction [unset\(\)](#), la fonction [isset\(\)](#) renverra FALSE.

```
<?php
$a = "test";
echo isset ($a); // TRUE
unset($a);
echo isset ($a); // FALSE ?>
```

Voir aussi [empty\(\)](#) et [unset\(\)](#).

[10.74.18 print_r](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [print_r](#) (mixed *expression*)

[PHP 4]

Cette fonction affiche des informations à propos d'une variable, de manière à ce qu'elle soit lisible. Pour une chaîne, un entier ou un double, la valeur sera elle-même affichée. Pour les tableaux, les valeurs seront présentées dans un format qui montre les clés et les valeurs. Une notation similaire est disponible pour les objets.

Comparer [print_r\(\)](#) et [var_dump\(\)](#).


```
<?php
$a = array (1, 2, array ("a", "b", "c"));
print_r ($a);
?>
```

10.74.19 [serialize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [serialize](#) (mixed *value*)
[PHP 3>= 3.0.5, PHP 4]

[serialize\(\)](#) retourne une chaîne contenant une représentation linéaire de *value*, pour stockage.

C'est une technique pratique pour stocker ou passer des valeurs de PHP entre scripts, sans perdre ni leur structure, ni leur type.

Pour récupérer une variable linéarisée, et retrouver une variable, utilisez [unserialize\(\)](#). [serialize\(\)](#) acceptent les types integer, double, string, array (multidimensionnels) et object (les propriétés des objets seront linéarisées, mais pas les méthodes).

Exemple avec serialize()

```
<?php
// $session_data contient un tableau multi-dimensionnel , avec les
// informations de session de l'utilisateur courant. On utilise serialize()
// pour les stocker dans une base de données
$conn = odbc_connect ("webdb", "php", "chicken");
$stmt = odbc_prepare ($conn,
    "UPDATE sessions SET data = ? WHERE id = ?");
$sqldata = array (serialize($session_data), $PHP_AUTH_USER);
if (!odbc_execute ($stmt, &$sqldata)) {
    $stmt = odbc_prepare($conn,
        "INSERT INTO sessions (id, data) VALUES(?, ?)");
    if (!odbc_execute($stmt, &$sqldata)) {
        /* Grosse bourde! Souffre et potasse! */
    }
}
?>
```

10.74.20 [settype](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [settype](#) (string *var*, string *type*)
[PHP 3, PHP 4]

[settype\(\)](#) modifie le type de la variable *var* en *type*.

Les valeurs possibles pour le paramètre *type* sont :

- "integer"
- "double"
- "string"

- "array"
- "object"

Renvoie TRUE en cas de succès, FALSE sinon.

Voir aussi [gettype\(\)](#).

10.74.21 strval

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [strval](#) (mixed *var*)
[PHP 3, PHP 4]

Retourne la valeur de la variable *var*, au format chaîne de caractères.

var peut être un scalaire. Vous ne pouvez pas utiliser la fonction [strval\(\)](#) avec des tableaux ou des objets.

Voir aussi [doubleval\(\)](#), [intval\(\)](#), [settype\(\)](#) et [9.8.6 Définition du type](#).

10.74.22 unserialize

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [unserialize](#) (string *str*)
[PHP 3>= 3.0.5, PHP 4]

[unserialize\(\)](#) prend une variable linéarisée (voir [serialize\(\)](#)) et la convertit en variable PHP. La valeur convertie est retournée par la fonction, et peut être de type integer, double, string, array ou object. Les objets linéarisés perdent leurs méthodes.

Exemple avec unserialize()

```
<?php
// Ici, on utilise unserialize\(\) pour charger les données de sessions
// depuis la base de données, dans $session_data. Cet exemple complète
// celui fourni avec serialize\(\).
$conn = odbc_connect ("webdb", "php", "chicken");
$stmt = odbc_prepare ($conn, "SELECT data FROM sessions WHERE id = ?");
$sqldata = array ($PHP_AUTH_USER);
if (!odbc_execute ($stmt, &$sqldata) || !odbc_fetch_into ($stmt, &$tmp)) {
    // si la préparation ou la lecture échoue, on crée un tableau vide
    $session_data = array();
} else {
    // les données sauvées sont dans $tmp[0].
    $session_data = unserialize ($tmp[0]);
if (!is_array ($session_data)) {
    // Erreur... initialisation à tableau vide
    $session_data = array();
}
}
?>
```

10.74.23 unset

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [unset](#)(mixed *var*, mixed *var*, ...)

[unset\(\)](#) détruit les variables *var*, et retourne TRUE.

Exemple avec unset()

```
<?php
// Destruction d'une seule variable
unset ($foo);
// Destruction d'un élément de tableau
unset ($bar['quux']);
// Destruction de plusieurs variables
unset ($foo1, $foo2, $foo3);
?>
```

Le comportement de [unset\(\)](#) à l'intérieur d'une fonction peut varier suivant le type de variable que vous voulez détruire.

Si une variable globale est détruite avec [unset\(\)](#) depuis une fonction, seule la variable locale sera détruite. La variable globale gardera la valeur acquise avant l'appel à [unset\(\)](#).

```
<?php
function destroy_foo() {
    global $foo;
    unset($foo);
}
$foo = 'bar';
destroy_foo();
echo $foo;
?>
```

L'exemple ci dessus affichera : bar

Si une variable qui est passée par référence est détruite à l'intérieur d'une fonction, seule la variable locale sera détruite. La variable globale conservera la dernière valeur qu'elle avait avant l'appel de [unset\(\)](#).

```
<?php
function foo(&$bar) {
    unset($bar);
    $bar = "bla";
}
$bar = 'truc';
echo "$bar\n";
foo($bar);
echo "$bar\n";
?>
```

L'exemple ci dessus va afficher : truc truc

Si une variable statique est détruite à l'intérieure d'une fonction [unset\(\)](#) détruira la référence à la variable statique, plutôt que la variable statique elle même.

```
<?php
function foo() {
```

```
static $a;
    $a++;
echo "$a\n";
unset($a);
}
foo();
foo();
foo();
?>
```

L'affichage du script ci-dessus donnera : 1 2 3

Si vous voulez détruire une variable globale, depuis une fonction, vous pouvez utiliser le tableau **\$GLOBALS** :

```
<?php
function foo() {
unset($GLOBALS['bar']);
}
$bar = "truc";
foo();
?>
```

Voir aussi [isset\(\)](#) et [empty\(\)](#).

10.74.24 var_dump

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [var_dump](#) (mixed *expression*)

[PHP 3>= 3.0.5, PHP 4]

Cette fonction retourne les informations structurées d'une variable, y compris son type et sa valeur. Les tableaux sont explorés récursivement, avec des indentations, pour mettre en valeur leur structure.

Comparez [var_dump\(\)](#) et [print_r\(\)](#).

```
<pre>
<?php
    $a = array (1, 2, array ("a", "b", "c"));
var_dump ($a);
?>
</pre>
```

10.75 WDDX

[\[Notes en ligne\]](#)

Ces fonctions doivent fonctionner avec l'aide de [WDDX](#).

Pour utiliser WDDX, you devez installer la librairie EXPAT (qui est fournie avec la distribution d'Apache 1.3.7 ou plus récent), et recompiler PHP avec `--with-xml` et `--enable-wddx`.

Notez bien que toutes les fonctions qui enregistrent des données, utilisent le premier élément d'un tableau pour savoir si ce tableau doit être enregistré sous la forme d'un tableau, ou d'une structure. Si le premier élément a une clé de type chaîne, le tableau sera enregistré sous la forme d'une structure, et sinon, sous la

forme d'un tableau.

Enregistrer une valeur simple

```
<?php
print wddx_serialize_value("Exemple de paquet de PHP à WDDX ", "Paquet PHP");
?>
```

Cet exemple va produire le résultat suivant :

```
<wddxPacket version='0.9'><header comment='Paquet PHP' ><data>
<string>Exemple de paquet de PHP à WDDX</string></data></wddxPacket>
```

Utilisation de paquets incrémentaux

```
<?php
$pi = 3.1415926;
$packet_id = wddx_packet_start("PHP");
wddx_add_vars($packet_id, "pi");
/* Supposons que $villes provient d'une base de données */
$cities = array("Paris", "Marseilles", "Lyon");
wddx_add_vars($packet_id, " villes ");
$packet = wddx_packet_end($packet_id);
print $packet;
?>
```

Cet exemple donnera :

```
<wddxPacket version='0.9'><header comment='PHP' ><data><struct>
<var name='pi'><number>3.1415926</number></var><var name='cities'>
<array length='3'><string>Paris</string><string>Marseilles</string>
<string>Lyon</string></array></var></struct></data></wddxPacket>
```

10.75.1 wddx_serialize_value

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wddx_serialize_value](#) (mixed *var*, string *comment*)
[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[wddx_serialize_value\(\)](#) sert à créer un paquet WDDX à partir d'une seule valeur. Cette fonction prend la valeur de *var*, et un argument optionnel *comment* qui apparaîtra dans l'entête du paquet, et retourne un paquet WDDX.

10.75.2 wddx_serialize_vars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wddx_serialize_vars](#) (mixed *var_name*, mixed ...)
[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[wddx_serialize_vars\(\)](#) sert à créer un paquet WDDX avec une structure qui contient la représentation des variables passées en arguments.

[wddx_serialize_vars\(\)](#) prend un nombre variable d'arguments, chacun d'entre eux pouvant être une chaîne contenant le nom d'une variable, ou un tableau de chaîne de nom de variable, ou même d'autres tableaux.

wddx_serialize_vars()

```
<?php
$a = 1;
$b = 5.5;
$c = array("bleu", "orange", "violet");
$d = "colors";
$clvars = array("c", "d");
print wddx_serialize_vars("a", "b", $clvars);
?>
```

L'exemple ci-dessus donnera : <wddxPacket version='0.9'><header><data><struct><var name='a'><number>1</number></var> <var name='b'><number>5.5</number></var><var name='c'><array length='3'><string>bleu</string><string>orange</string><string>violet</string></array></var><var name='d'><string>colors</string></var></struct></data></wddxPacket>

10.75.3 wddx_packet_start

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [wddx_packet_start](#)(string *comment*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[wddx_packet_start\(\)](#) sert à créer un nouveau paquet WDDX, pour pouvoir y faire des ajouts incrémentaux de variables. Cette fonction prend un argument optionnel *comment* et retourne un identifiant de paquet, qui servira à d'autres fonctions. Elle va automatiquement créer une définition de structure dans le paquet, pour accueillir des variables.

10.75.4 wddx_packet_end

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [wddx_packet_end](#)(int *packet_id*)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[wddx_packet_end\(\)](#) clos un paquet WDDX repéré par son identifiant *packet_id*.

10.75.5 wddx_add_vars

[\[Notes en ligne\]](#) [\[Exemples\]](#)

[wddx_add_vars](#)(int *packet_id*, mixed *name_var*, mixed ...)

[PHP 3>= 3.0.7, PHP 4 >= 4.0b2]

[wddx_add_vars\(\)](#) sert à enregistrer les variables passées en argument, et les ajouter au paquet repéré par son identifiant *packet_id*. Les variables enregistrées sont spécifiées de la même façon que pour

[wddx_serialize_vars\(\)](#).

[10.75.6 wddx_deserialize](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [wddx_deserialize](#) (string *packet*)

[PHP 3 >= 3.0.7, PHP 4 >= 4.0b2]

[wddx_deserialize\(\)](#) prend une chaîne *packet* et la lit. Cette fonction retourne un résultat qui peut être une chaîne, un nombre ou un tableau. Notez que les structures sont lues sous la forme de tableau associatifs.

[10.76 Analyseur syntaxique XML](#)

[\[Notes en ligne\]](#)

[10.76.1 Introduction](#)

[\[Notes en ligne\]](#)

[10.76.1.1 A propos de XML](#)

[\[Notes en ligne\]](#)

Le langage XML (eXtensible Markup Language (Langage à Balises Etendu)) est un format structuré de données pour les échanges sur le web. C'est un standard défini par le consortium World Wide Web (W3C). Plus d'informations à propos du XML et des technologies afférentes sont accessibles (en anglais) <http://www.w3.org/XML/>.

[10.76.1.2 Installation](#)

[\[Notes en ligne\]](#)

Cette extension de PHP utilise *expat*, disponible à <http://www.jclark.com/xml/>. Le fichier Makefile livré avec *expat* ne construit pas par défaut de librairie : il faut utiliser la ligne suivante :

```
libexpat.a: $(OBJJS)
ar -rc $ $(OBJJS)
ranlib $
```

Les sources RPM de *expat* sont disponibles à <http://www.guardian.no/~ssb/phpxml.html>.

Notez que si vous utilisez Apache-1.3.7 ou plus récent, vous disposez déjà de la librairie *expat*. Configurez simplement PHP avec `--with-xml` (sans aucune autre information) et la librairie *expat* d'Apache sera automatiquement utilisée.

Sous UNIX, lancez la configuration de PHP avec l'option `--with-xml`, la librairie *expat* étant installée là où votre compilateur peut la trouver. Si vous compilez PHP comme module de PHP 1.3.9 ou plus récent, PHP utilisera automatiquement le module *expat* livré avec Apache. Il vous faudra peut être fixer les valeurs des variables d'environnement *CPPFLAGS* et *LDFLAGS*, si vous avez fait une installation exotique. Compilez PHP. *Tada!* C'est fait !

[10.76.1.3 A propos de cette extension :](#)

[\[Notes en ligne\]](#)

Cette extension PHP supporte la librairie *expat* de James Clark sous PHP. Cela vous permettra d'analyser

mais pas de valider les documents XML. Il supporte trois types de codage différents, disponibles aussi sous PHP: US-ASCII, ISO-8859-1 et UTF-8. UTF-16 n'est pas supporté.

Cette extension vous permet de créer des [10.76.3 xml_parser_create](#) puis de définir des *points d'entrée* pour chaque événement XML. Les analyseurs XML disposent de quelques [10.76.20 xml_parser_set_option](#). Les gestionnaires d'événements XML sont:

Fonction PHP de configuration du gestionnaire	Description de l'événement
xml_set_element_handler()	Un événement est généré à chaque fois que l'analyseur XML rencontre une balise de début ou de fin. Deux gestionnaires sont disponibles : un pour le début, et un pour la fin. @tab
xml_set_character_data_handler() @tab "Character data" correspond grosso modo à tout ce qui n'est pas une balise XML, y compris les espaces entre les balises. Notez bien que l'analyseur XML n'ajoute ou n'efface aucun espace, et que c'est à l'application (c'est à dire vous) de décider de la signification de ces espaces. @tab	
xml_set_processing_instruction_handler() @tab Les programmeurs PHP sont habitués aux instructions exécutables (processing instructions ou PIs). <?php ?> est une instruction exécutable où php est appelé programme cible. Ces instructions sont gérées de manière spécifiques, (sauf le programme cible, qui est réservé à XML). @tab	
xml_set_default_handler()	Tout ce qui n'a pas trouvé de gestionnaire est transmis au gestionnaire par défaut. Vous retrouverez par exemple, les déclarations de type de document dans ce gestionnaire. @tab
xml_set_unparsed_entity_decl_handler() @tab Ce gestionnaire est appelé pour gérer les déclaration des entités non analysés. @tab	
xml_set_notation_decl_handler() @tab Ce gestionnaire est appelé pour gérer les notations. @tab	
xml_set_external_entity_ref_handler() @tab Ce gestionnaire est appelé lorsque l'analyseur XML trouve une référence à un fichier externe. Cela peut être un fichier, ou une URL. Reportez vous à 10.76.2.3 XML Entité externe pour un exemple. @tab	

[10.76.1.4 Problèmes de casse](#)

[\[Notes en ligne\]](#)

Les fonctions de gestion des balises peuvent rencontrer des balises en minuscule, majuscule ou encore dans un mélange des deux. En XML, la procédure standard est d' "identifier les séquences de caractère qui ne sont pas reconnues comme majuscule, et de les remplacer par leur équivalent majuscule". En d'autres termes, XML met toutes lettres en majuscules.

Par défaut, tous les noms des éléments qui sont transmis aux fonctions de gestion sont mises en majuscule.

Ce comportement est contrôlé par l'analyseur XML, et peut être lu et modifié avec les fonctions respectives [xml_parser_get_option\(\)](#) et [xml_parser_set_option\(\)](#), respectivement.

10.76.1.5 Error Codes

[\[Notes en ligne\]](#)

Les constantes suivantes sont définies comme des codes d'erreurs XML : (retournée par [xml_parse\(\)](#))

- XML_ERROR_NONE
- XML_ERROR_NO_MEMORY
- XML_ERROR_SYNTAX
- XML_ERROR_NO_ELEMENTS
- XML_ERROR_INVALID_TOKEN
- XML_ERROR_UNCLOSED_TOKEN
- XML_ERROR_PARTIAL_CHAR
- XML_ERROR_TAG_MISMATCH
- XML_ERROR_DUPLICATE_ATTRIBUTE
- XML_ERROR_JUNK_AFTER_DOC_ELEMENT
- XML_ERROR_PARAM_ENTITY_REF
- XML_ERROR_UNDEFINED_ENTITY
- XML_ERROR_RECURSIVE_ENTITY_REF
- XML_ERROR_ASYNC_ENTITY
- XML_ERROR_BAD_CHAR_REF
- XML_ERROR_BINARY_ENTITY_REF
- XML_ERROR_ATTRIBUTE_EXTERNAL_ENTITY_REF
- XML_ERROR_MISPLACED_XML_PI
- XML_ERROR_UNKNOWN_ENCODING
- XML_ERROR_INCORRECT_ENCODING
- XML_ERROR_UNCLOSED_CDATA_SECTION
- XML_ERROR_EXTERNAL_ENTITY_HANDLING

10.76.1.6 Codage des caractères

[\[Notes en ligne\]](#)

L'extension XML de PHP supporte les caractères [Unicode](#) grâce à différents codages. Il y a deux types de codages de caractères : le codage à la source et le codage à la cible. PHP utilise le UTF-8 comme représentation interne.

L'encodage à la source est effectué lors de [10.76.12 xml_parse](#) du fichier par XML. Lors de la [10.76.3 xml_parser_create](#), un type de codage à la source doit être spécifié (et il ne pourra plus être modifié jusqu'à la destruction de l'analyseur). Les codages supportés sont : ISO-8859-1, US-ASCII et UTF-8. Les deux derniers sont des codages à un seul octet, c'est à dire que les caractères sont représentés sur un seul octet. UTF-8 peut représenter des caractères composés par un nombre variable de bits (jusqu'à 21), allant de 1 à quatre octets. Le codage par défaut utilisé par PHP ISO-8859-1.

Le codage à la cible est effectué lorsque PHP transfère les données aux gestionnaires XML. Lorsqu'un analyseur est créé, le codage à la cible est spécifié de la même façon que le codage à la source, mais il peut être modifié à tout moment. Le codage à la cible affectera les balises, tout comme les données brutes, et les noms des instructions exécutables.

Si l'analyseur XML rencontre un caractère qu'il ne connaît pas (hors limite, par exemple), il retournera une erreur.

Si PHP rencontre un caractère dans le document XML analysé, qu'il ne peut pas représenter dans le codage à la cible choisi, le caractère sera remplacé par un point d'interrogation (cette attitude est susceptible de changer ultérieurement).

10.76.2 Quelques exemples

[\[Notes en ligne\]](#)

Voici une liste d'exemple de code PHP qui analyse un document XML.

10.76.2.1 Exemple de structure XML

[\[Notes en ligne\]](#)

Ce premier exemple affiche la structure de l'élément de début dans un document avec indentation.

Afficher une structure XML

```
<?php
$file = "data.xml";
$depth = array();
function startElement($parser, $name, $attrs) {
    global $depth;
    for ($i = 0; $i < $depth[$parser]; $i++) {
        print "  ";
    }
    print "$name\n";
    $depth[$parser]++;
}
function endElement($parser, $name) {
    global $depth;
    $depth[$parser]--;
}
$xml_parser = xml_parser_create();
xml_set_element_handler($xml_parser, "startElement", "endElement");
if (!$fp = fopen($file, "r")) {
    die("could not open XML input");
}
while ($data = fread($fp, 4096)) {
    if (!xml_parse($xml_parser, $data, feof($fp))) {
        die(sprintf("XML error: %s at line %d",
            xml_error_string(xml_get_error_code($xml_parser)),
            xml_get_current_line_number($xml_parser)));
    }
}
xml_parser_free($xml_parser);
?>
```

10.76.2.2 XML Transtypage XML -> HTML

[\[Notes en ligne\]](#)

XML Transtypage XML -> HTML

Cet exemple remplace les balises XML d'un document par des balises HTML. Les éléments inconnus seront ignorés. Bien entendu, cet exemple sera appliqué à un type

précis de fichiers XML.

```
<?php
$file = "data.xml";
$map_array = array(
    "BOLD"      => "B",
    "EMPHASIS" => "I",
    "LITERAL"  => "TT"
);
function startElement($parser, $name, $attrs) {
    global $map_array;
    if ($htmltag = $map_array[$name]) {
        print "<$htmltag>";
    }
}
function endElement($parser, $name) {
    global $map_array;
    if ($htmltag = $map_array[$name]) {
        print "</$htmltag>";
    }
}
function characterData($parser, $data) {
    print $data;
}
$xml_parser = xml_parser_create();
// use case-folding so we are sure to find the tag in $map_array
xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING, TRUE);
xml_set_element_handler($xml_parser, "startElement", "endElement");
xml_set_character_data_handler($xml_parser, "characterData");
if (!$fp = fopen($file, "r")) {
    die("could not open XML input");
}
while ($data = fread($fp, 4096)) {
    if (!xml_parse($xml_parser, $data, feof($fp))) {
        die(sprintf("XML error: %s at line %d",
            xml_error_string(xml_get_error_code($xml_parser)),
            xml_get_current_line_number($xml_parser)));
    }
}
xml_parser_free($xml_parser);
?>
```

10.76.2.3 XML Entité externe

[\[Notes en ligne\]](#)

Cet exemple exploite les références externes de XML : il est possible d'utiliser un gestionnaire d'entité externe pour inclure et analyser les documents, tous comme les instructions exécutables peuvent servir à inclure et analyser d'autres documents, et aussi fournir une indication de confiance (voir plus bas). Le document XML qui est utilisé dans cet exemple est fourni plus loin dans l'exemple ('xmltest.xml' et 'xmltest2.xml').

Entité externe

```

<?php
$file = "xmltest.xml";
function trustedFile($file) {
    // only trust local files owned by ourselves
    if (!ereg("^[a-z]+://", $file)
        && fileowner($file) == getmyuid()) {
    return TRUE;
    }
    return FALSE;
}
function startElement($parser, $name, $attribs) {
    print "<";
    if ($name != "font") {
        print "<font color=\"";
        if ($name == "b") {
            print "0000cc\"";
        } else {
            print "009900\"";
        }
        print ">$name</font>";
    }
    if (sizeof($attribs)) {
        while (list($k, $v) = each($attribs)) {
            print " <font color=\"";
            if ($k == "b") {
                print "0000cc\"";
            } else {
                print "009900\"";
            }
            print ">$k</font>=";
            if ($v != "b") {
                print "<font color=\"";
                if ($v == "b") {
                    print "0000cc\"";
                } else {
                    print "009900\"";
                }
                print ">$v</font>\"";
            }
        }
    }
    print ">";
}
function endElement($parser, $name) {
    print "<";
    if ($name != "font") {
        print "<font color=\"";
        if ($name == "b") {
            print "0000cc\"";
        } else {
            print "009900\"";
        }
        print ">$name</font>";
    }
    print ">";
}
function characterData($parser, $data) {
    print "<B>$data</B>";
}
function PIHandler($parser, $target, $data) {
    switch (strtolower($target)) {
        case "php":
            global $parser_file;
            // If the parsed document is "trusted", we say it is safe
            // to execute PHP code inside it. If not, display the code
            // instead.
            if (trustedFile($parser_file[$parser])) {
                eval($data);
            } else {
                printf("Code PHP peu sûr : <B>%s</B>",
                    htmlspecialchars($data));
            }
            break;
    }
}
function defaultHandler($parser, $data) {
    if (substr($data, 0, 1) == "&" && substr($data, -1, 1) == ";") {
        printf('<font color="#aa00aa">%s</font>',
            htmlspecialchars($data));
    } else {
        printf('<font size="-1">%s</font>',
            htmlspecialchars($data));
    }
}
function externalEntityRefHandler($parser, $openEntityNames, $base, $systemId,
    $publicId) {
    if ($systemId) {
        if (!list($parser, $fp) = new_xml_parser($systemId)) {
            printf("Could not open entity %s at %s\n", $openEntityNames,
                $systemId);
        }
        return FALSE;
    }
    while ($data = fread($fp, 4096)) {
        if (!xml_parse($parser, $data, feof($fp))) {
            printf("XML error: %s at line %d while parsing entity %s\n",

```

```

xml_error_string(xml_get_error_code($parser)),
xml_get_current_line_number($parser), $openEntityNames);
xml_parser_free($parser);
return FALSE;
    }
}
xml_parser_free($parser);
return TRUE;
}
return FALSE;
}
function new_xml_parser($file) {
global $parser_file;
$xml_parser = xml_parser_create();
xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING, 1);
xml_set_element_handler($xml_parser, "startElement", "endElement");
xml_set_character_data_handler($xml_parser, "characterData");
xml_set_processing_instruction_handler($xml_parser, "PIHandler");
xml_set_default_handler($xml_parser, "defaultHandler");
xml_set_external_entity_ref_handler($xml_parser, "externalEntityRefHandler");
if (!$fp = @fopen($file, "r")) {
return FALSE;
}
if (!is_array($parser_file)) {
settype($parser_file, "array");
}
$parser_file[$xml_parser] = $file;
return array($xml_parser, $fp);
}
if (!(list($xml_parser, $fp) = new_xml_parser($file))) {
die("could not open XML input");
}
print "<pre>";
while ($data = fread($fp, 4096)) {
if (!xml_parse($xml_parser, $data, feof($fp))) {
die(sprintf("XML error: %s at line %d\n",
xml_error_string(xml_get_error_code($xml_parser)),
xml_get_current_line_number($xml_parser)));
}
}
print "</pre>";
print "parse complete\n";
xml_parser_free($xml_parser);
?>

```

xmltest.xml

```

<?xml version='1.0'?>
<!DOCTYPE chapter SYSTEM "/just/a/test.dtd" [
<!ENTITY plainEntity "FOO entity">
<!ENTITY systemEntity SYSTEM "xmltest2.xml">
?>
<chapter>
<TITLE>Title &plainEntity;</TITLE>
<para>
<informaltable>
<tgroup cols="3">
<tbody>
<row><entry>a</entry><entry morerows="1">b</entry><entry>c</entry></row>

```

```

        <row><entry>a2</entry><entry>c2</entry></row>
        <row><entry>a3</entry><entry>b3</entry><entry>c3</entry></row>
    </tbody>
</tgroup>
</informaltable>
</para>
&systemEntity;
<sect1 id="about">
    <title>About this Document</title>
    <para>
        <!-- this is a comment -->
        <?php print 'Hi!  This is PHP version '.phpversion(); ">
    </para>
</sect1>
</chapter>

```

This file is included from `xmltest.xml`:

xmltest2.xml

```

<?xml version="1.0" ?>
<!DOCTYPE foo [
<!ENTITY testEnt "test entity">
?>
<foo>
    <element attrib="value"?>
        &testEnt;
        <?php print "This is some more PHP code being executed."; ?>
</foo>

```

[10.76.3 xml_parser_create](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parser_create](#)(string *encoding*)

[PHP 3>= 3.0.6, PHP 4]

encoding (optional)

- Le codage de caractère de l'analyseur : les codages suivants sont supportés :
 - ◆ ISO-8859-1 (par défaut)
 - ◆ US-ASCII
 - ◆ UTF-8

Cette fonction crée un analyseur XML et retourne une référence sur cet analyseur pour qu'il puisse être utilisé ultérieurement par d'autres fonctions XML. Retourne FALSE en cas d'erreur.

[10.76.4 xml_set_object](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [xml_set_object](#)(int *parser*, object *&object*)

[PHP 4 >= 4.0b4]

Cette fonction rend l'analyseur *parser* utilisable depuis un objet. Toutes les méthodes de callback, affectées par [xml_set_element_handler\(\)](#), seront les méthodes de cet objet.

```
<?php
class xml {
var $parser;
function xml() {
    $this->parser = xml_parser_create();
    xml_set_object($this->parser,&$this);
    xml_set_element_handler($this->parser,"tag_open","tag_close");
    xml_set_character_data_handler($this->parser,"cdata");
}
function parse($data) {
    xml_parse($this->parser,$data);
}
function tag_open($parser,$tag,$attributes) {
    var_dump($parser,$tag,$attributes);
}
function cdata($parser,$cdata) {
    var_dump($parser,$cdata);
}
function tag_close($parser,$tag) {
    var_dump($parser,$tag);
}
} // Fin de la classe xml
$xml_parser = new xml();
$xml_parser->parse("<A ID=\"bonjour\">PHP</?>");
?>
```

10.76.5 [xml_set_element_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_element_handler](#) (int *parser*, string *startElementHandler* , string *endElementHandler*)

[PHP 3>= 3.0.6, PHP 4]

Affecte les gestionnaires de début et de fin de l'analyseur XML *parser*. *startElementHandler* et *endElementHandler* sont des chaînes qui contiennent les noms de fonctions qui existent lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

La fonction *startElementHandler* doit accepter trois paramètres: *startElementHandler* (int *parser*, string *name*, array *attribs*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.
- name*
- Le deuxième paramètre, *name*, contient le nom de l'élément qui a provoqué l'appel du gestionnaire. Si l'analyseur gère la [10.76.1.4 Problèmes de casse](#), cet élément sera en majuscule.
- attribs*
- Le troisième paramètre, *attribs*, contient un tableau associatif avec les attributs de l'éléments (si il en existe). Les clés de ce tableau seront les noms des attributs, et les valeurs seront les valeurs correspondantes des attributs. Les noms des attributs seront mis en majuscule si l'analyseur gère la [10.76.1.4 Problèmes de casse](#). Les valeurs des attributs seront intouchées. L'ordre original des attributs peut être retrouvé en passant en revue le tableau *attribs*, avec la fonction

[each\(\)](#). La première clé sera la première clé du tableau.

La fonction ***endElementHandler*** doit accepter deux paramètres: ***endElementHandler*** (int ***parser***, string ***name***)

parser

- Le premier paramètre, ***parser***, est une référence sur l'analyseur XML qui appelle cette fonction.
- Le second paramètre, ***name***, contient le nom de l'élément qui a provoqué l'appel du gestionnaire. Si l'analyseur gère la [10.76.1.4 Problèmes de casse](#), cet élément sera en majuscule.

Si un gestionnaire reçoit une chaîne vide, ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si ***parser*** n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire. Reportez vous à [xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

[10.76.6 xml_set_character_data_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_character_data_handler](#) (int ***parser***, string ***handler***)

[PHP 3>= 3.0.6, PHP 4]

Affecte les gestionnaires de début et de fin de l'analyseur XML ***parser***. ***handler*** est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer ***parser***.

La fonction ***handler*** doit accepter deux paramètres: ***handler*** (int ***parser***, string ***data***)

parser

- Le premier paramètre, ***parser***, est une référence sur l'analyseur XML qui appelle cette fonction.
- Le second paramètre, ***data***, contient les caractères sous la forme d'une chaîne.

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si ***parser*** n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire. Reportez vous à [xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

[10.76.7 xml_set_processing_instruction_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_processing_instruction_handler](#) (int ***parser***, string ***handler***)

[PHP 3>= 3.0.6, PHP 4]

Affecte les gestionnaires d'instructions exécutables de l'analyseur XML ***parser***. ***handler*** est une chaîne qui

contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*. Une instruction exécutable a la forme suivante :

```
<?
target
data
```

Vous pouvez mettre du code PHP entre ces balises, mais soyez conscient d'une des limitations des instructions exécutables de XML : la balise de fin d'instruction exécutable (?>) ne peut être échappée, ce qui fait que cette séquence NE DOIT JAMAIS apparaître dans le code PHP placé dans le document PHP. Si un tel texte apparaît, la balise de fin d'instruction exécutable sera reconnue, et le reste du code sera considéré comme des données brutes (et donc, pas exécutées).

La fonction *handler* doit accepter trois paramètres: *handler* (int *parser*, string *target*, string *data*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

target

- Le second paramètre, *target*, contient l'application cible.

data

- Le troisième paramètre, *data*, contient le code sous la forme d'une chaîne.

Si un gestionnaire reçoit une chaîne vide, ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire. Reportez vous à [xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

10.76.8 [xml_set_default_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_default_handler](#) (int *parser*, string *handler*)

[PHP 3>= 3.0.6, PHP 4]

Affecte le gestionnaire par défaut de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

La fonction *handler* doit accepter deux paramètres: *handler* (int *parser*, string *data*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.

data

- Le second paramètre, *data*, contient les caractères sous la forme d'une chaîne. Cela peut être une déclaration XML, un type de document, une entité ou d'autre données pour qui aucun gestionnaire n'est prévu.

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire. Reportez vous à

[xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

[10.76.9 xml_set_unparsed_entity_decl_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_unparsed_entity_decl_handler](#) (int *parser*, string *handler*)

[PHP 3>= 3.0.6, PHP 4]

Affecte les gestionnaires d'entité non déclaré de l'analyseur XML *parser*. *handler* est une chaîne qui contient le nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.

Ce gestionnaire sera appelé si l'analyseur XML rencontre une déclaration d'entité externe avec une déclaration de NDATA, comme suit :

```
<!ENTITY name {publicId | systemId}  
NDATA notationName
```

Reportez vous à la section [des spécifications XML 1.0](#) pour connaître les notations des entités externes.

La fonction *handler* doit accepter six paramètres: *handler* (int *parser*, string *entityName*, string *base*, string *systemId*, string *publicId*, string *notationName*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.
- entityName*
- Le nom de l'entité qui va être définie
- base*
- La meilleure base de résolution de l'identifiant système de cette entité externe. Actuellement, ce paramètre est toujours une chaîne vide.
- systemId*
- Identifiant système pour cet entité externe.
- publicId*
- Identifiant public pour cet entité externe.
- notationName*
- Nom de la notation de cette entité. (Voir [xml_set_notation_decl_handler\(\)](#)).

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé.

Retourne TRUE si le gestionnaire est actif, et FALSE sinon, ou si *parser* n'est pas un analyseur.

Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire. Reportez vous à [xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

[10.76.10 xml_set_notation_decl_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_notation_decl_handler](#) (int *parser*, string *handler*)

[PHP 3>= 3.0.6, PHP 4]

Affecte les gestionnaires de début et de fin de l'analyseur XML *parser*. *handler* est une chaîne qui contient le

nom d'une fonction qui existe lorsque [xml_parse\(\)](#) est appelé pour créer *parser*.
Une notation est une partie du DTD du document, qui a le format suivant :

```
<!NOTATION name
{ systemId | publicId?>
```

Reportez vous à la section [des spécifications XML 1.0](#) pour connaître les notations des entités externes.

La fonction *handler* doit accepter cinq paramètres: *handler* (int *parser*, string *notationName*, string *base*, string *systemId*, string *publicId*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.
- *notationName*
- Le nom de la notation, *name*, comme précisé dans le format de notation ci dessus.
- *base*
- La meilleure base de résolution de l'identifiant système de cette entité externe. Actuellement, ce paramètre est toujours une chaîne vide.
- *systemId*
- Identifiant système pour cette entité externe.
- *publicId*
- Identifiant public pour cet entité externe.

Si un gestionnaire reçoit une chaîne vide ou FALSE, c'est qu'il est en train d'être désactivé.

Retourne TRUE si le gestionnaire est actif, et FALSE sinon ou si *parser* n'est pas un analyseur.

Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaires. Reportez vous à [xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

[10.76.11 xml_set_external_entity_ref_handler](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_set_external_entity_ref_handler](#) (int *parser*, string *handler*)

[PHP 3>= 3.0.6, PHP 4]

Fixe le gestionnaire d'entité externe de l'analyseur XML *parser*. *handler* et *endElementHandler* sont des chaînes qui contiennent les noms de fonction qui existent lorsque [xml_parse\(\)](#) est appelé pour créer le *parser*.

La fonction *handler* doit accepter 5 paramètres, et retourner un entier. Si la valeur retournée par le gestionnaire est FALSE (comme par exemple si aucune valeur n'est retournée), l'analyseur XML s'arrêtera, et

la fonction [xml_get_error_code\(\)](#) retournera XML_ERROR_EXTERNAL_ENTITY_HANDLING. int

handler (int *parser*, string *openEntityNames*, string *base*, string *systemId*, string *publicId*)

parser

- Le premier paramètre, *parser*, est une référence sur l'analyseur XML qui appelle cette fonction.
- *openEntityNames*
- Le deuxième paramètre, *openEntityNames*, est i.e. liste de noms d'entité, séparés par des espaces. Ces entités sont accessibles à l'analyse par cet entité (y compris le nom de l'entité référencé).

base

- La meilleure base de résolution de l'identifiant système de cet entité externe. Actuellement, ce paramètre est toujours une chaîne vide.

systemId

- Identifiant système pour cet entité externe.

publicId

- Le cinquième paramètre, ***publicId***, est l'identifiant public, comme spécifié dans la déclaration d'entité, ou un chaîne vide, si aucune déclaration n'a été spécifiée. L'espace dans l'identifiant public sera normalisé comme spécifié dans les spécifications XML.

Si un gestionnaire reçoit une chaîne vide, ou FALSE, c'est qu'il est en train d'être désactivé. Retourne TRUE si le gestionnaire est actif, et FALSE sinon ou si ***parser*** n'est pas un analyseur. Il n'est pas pour l'instant possible d'utiliser des objets pour servir de gestionnaire. Reportez vous à [xml_set_object\(\)](#) pour utiliser l'analyseur XML depuis un objet.

10.76.12 xml_parse

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parse](#)(int ***parser***, string ***data***, int ***isFinal***)
[PHP 3>= 3.0.6, PHP 4]

parser

- une référence sur l'analyseur XML à utiliser.

data

- Une partie des données à analyser. Un document peut être analysé morceau par morceau, en appelant [xml_parse\(\)](#) plusieurs fois, tant que le paramètre ***isFinal*** est mis à TRUE pour le dernier morceau. ***isFinal*** (optional)
- Si il vaut TRUE, ***data*** est la dernière partie à analyser.

Lorsqu'un document XML est analysé, les gestionnaires d'événements sont appelés aussi souvent que nécessaire, et retournent TRUE ou FALSE.

TRUE est retourné lorsque l'analyse a été concluante, et FALSE en cas d'échec, ou si ***parser*** n'est pas un analyseur valide. Lors d'un échec d'analyse, la cause de l'erreur peut être obtenue grâce aux fonctions [xml_get_error_code\(\)](#), [xml_error_string\(\)](#), [xml_get_current_line_number\(\)](#), [xml_get_current_column_number\(\)](#) et [xml_get_current_byte_index\(\)](#).

10.76.13 xml_get_error_code

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_error_code](#)(int ***parser***)
[PHP 3>= 3.0.6, PHP 4]

parser

- Cette fonction retourne FALSE si ***parser*** n'est pas valide, ou sinon, retourne le numéro de colonne courante de la ligne courante de l'analyseur, qui correspond à la position d'analyse courante de

l'analyseur XML.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de colonne courante de la ligne courante de l'analyseur, qui correspond à la position d'analyse courante de l'analyseur XML.

[10.76.14 xml_error_string](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [xml_error_string](#) (int *code*)
[PHP 3>= 3.0.6, PHP 4]

code

- Un message d'erreur, issu de [xml_get_error_code\(\)](#).

Retourne la chaîne avec un message textuel, décrivant l'erreur *code*, ou FALSE si aucune description n'a été trouvée.

[10.76.15 xml_get_current_line_number](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_current_line_number](#) (int *parser*)
[PHP 3>= 3.0.6, PHP 4]

parser

- Une référence sur un analyseur XML valide.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de la ligne en cours d'analyse.

[10.76.16 xml_get_current_column_number](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_current_column_number](#) (int *parser*)
[PHP 3>= 3.0.6, PHP 4]

parser

- Une référence sur un analyseur XML valide.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne le numéro de colonne courante de la ligne courante de l'analyseur, qui correspond à la position d'analyse courante de l'analyseur XML.

[10.76.17 xml_get_current_byte_index](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_get_current_byte_index](#)(int *parser*)

[PHP 3>= 3.0.6, PHP 4]

parser

- Une référence sur un analyseur XML valide.

Cette fonction retourne FALSE si *parser* n'est pas valide, ou sinon, retourne l'index de l'octet d'analyse courante de l'analyseur XML.

[10.76.18 xml_parse_into_struct](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parse_into_struct](#)(int *parser*, string *data*, array *&values*, array *&index*)

[PHP 3>= 3.0.8, PHP 4]

[xml_parse_into_struct\(\)](#) analyse le fichier XML *data*, et le place dans deux tableaux : le premier *index* contient des pointeurs sur la position des valeurs correspondantes dans le tableau *values* array. Ces deux paramètres sont passés par références.

Ci dessous, vous trouverez un exemple qui illustre la structure des deux tableaux générés par la fonction. On utilise une balise simple note, placée dans une autre balise para. On analyse le tout, et on affiche la structure générée :

```
<?php
$simple = "<para><note>simple note</note></para>";
$p = xml_parser_create();
xml_parse_into_struct($p,$simple,$vals,$index);
xml_parser_free($p);
echo "Tableau d'index\n";
print_r($index);
echo "\nTableau de valeurs\n";
print_r($vals);
?>
```

```
Lors de l'exécution du code, l'affichage sera : Tableau d'index Array ( [PARA] => Array (
[0] => 0 [1] => 2 ) [NOTE] => Array ( [0] => 1 ) ) Tableau de valeurs
Array ( [0] => Array ( [tag] => PARA [type] => open [level] => 1 ) [1]
=> Array ( [tag] => NOTE [type] => complete [level] => 2 [value] =>
simple note ) [2] => Array ( [tag] => PARA [type] => close [level] => 1
) )
```

L'analyse événementielle (comme celle de expat), peut se révéler complexe lorsque le document XML est complexe. Cette fonction ne génère pas d'objet de type DOM, mais il génère plutôt des structures qui peuvent être parcourues à la façon d'un arbre. Considérons le fichier suivant, qui représente une petite base de données XML :

molddb.xml – Petite base de données moléculaire

```
<?xml version="1.0"?>
```

```

<moldb>
  <molecule>
    <name>Alanine</name>
    <symbol>ala</symbol>
    <code>A</code>
    <type>hydrophobic</type>
  </molecule>
  <molecule>
    <name>Lysine</name>
    <symbol>lys</symbol>
    <code>K</code>
    <type>charged</type>
  </molecule>
</moldb>

```

Et maintenant, un code qui analyse le document, et génère les objet ad hoc :

parsemoldb.php – analyse moldb.xml et crée un tableau d'objet moléculaires

```

<?php
class AminoAcid {
var $name; // nom de l'acide aminé
var $symbol; // symbole en trois lettres
var $code; // code en une lettre
var $type; // hydrophobe, chargé ou neutre
function AminoAcid ($aa) {
foreach ($aa as $k->$v)
    $this->$k = $aa[$k];
}
}

function readDatabase($filename) {
    // read the xml database of aminoacids
    $data = implode("",file($filename));
    $parser = xml_parser_create();
    xml_parser_set_option($parser,XML_OPTION_CASE_FOLDING,0);
    xml_parser_set_option($parser,XML_OPTION_SKIP_WHITE,1);
    xml_parse_into_struct($parser,$data,&$values,&$tags);
    xml_parser_free($parser);
    // parcourt les structures
    foreach ($tags as $key->$val) {
    if ($key == "molecule") {
        $molranges = $val;
        // chaque paire contigue sont les définitions supérieures
        // et inférieures de la molécule
        for ($i=0; $i < count($molranges); $i+=2) {
            $offset = $molranges[$i] + 1;
            $len = $molranges[$i + 1] - $offset;
            $tdb[] = parseMol(array_slice($values, $offset, $len));
        }
    } else {
        continue;
    }
    }
    return $tdb;
}

function parseMol($mvalues) {
    for ($i=0; $i < count($mvalues); $i++)
        $mol[$mvalues[$i]["tag"]] = $mvalues[$i]["value"];
    return new AminoAcid($mol);
}

$db = readDatabase("moldb.xml");
echo "*** Database of AminoAcid objects:\n";

```

```
print_r($db);
?>
```

Après exécution de ``parsemolddb.php'`, la variable `$db` contient un tableau d'objets AminoAcid, et l'affichage le confirme : `** Database of AminoAcid objects: Array ([0] => aminoacid Object ([name] => Alanine [symbol] => ala [code] => A [type] => hydrophobic) [1] => aminoacid Object ([name] => Lysine [symbol] => lys [code] => K [type] => charged))`

10.76.19 xml_parser_free

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [xml_parser_free](#) (int *parser*)
[PHP 3>= 3.0.6, PHP 4]

parser

- Une référence sur un analyseur XML.

Cette fonction retourne FALSE si *parser* n'est pas une référence valide, ou sinon, détruit l'analyseur et retourne TRUE.

10.76.20 xml_parser_set_option

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [xml_parser_set_option](#) (int *parser*, int *option*, mixed *value*)
[PHP 3>= 3.0.6, PHP 4]

parser

- Une référence vers un analyseur XML.

option

- L'option à modifier. Voir ci dessous :

value

- La nouvelle valeur de l'option.

Cette fonction retourne FALSE si *parser* n'est pas une référence valide sur un analyseur XML, ou si l'option n'a pas pu être modifiée. Sinon, l'option est effectivement modifiée, et la fonction retourne TRUE.

Les options suivantes sont disponibles :

Option	Type de données	Description
XML_OPTION_CASE_FOLDING	entier	Contrôle la gestion de la 10.76.1.4 Problèmes de casse des balises de cet analyseur XML. Par défaut, activé. @tab

XML_OPTION_TARGET_ENCODING	string	Modifie le 10.76.1.6 Codage des caractères utilisé par cet analyseur XML. Par défaut, c'est celui qui a été spécifié lors de l'appel de xml_parser_create() . Les codages supportés sont ISO-8859-1, US-ASCII et UTF-8. @tab
----------------------------	--------	--

[10.76.21 xml_parser_get_option](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [xml_parser_get_option](#)(int *parser*, int *option*)
[PHP 3>= 3.0.6, PHP 4]

parser

- Une référence sur un analyseur XML valide.

option

- L'option demandée. Reportez vous à [xml_parser_set_option\(\)](#) pour avoir la liste des options disponibles.

[xml_parser_get_option\(\)](#) retourne FALSE si *parser* n'est pas valide, ou sinon, retourne la valeur de l'option demandée.

Reportez vous à [xml_parser_set_option\(\)](#) pour avoir la liste des options disponibles.

[10.76.22 utf8_decode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [utf8_decode](#)(string *data*)
[PHP 3>= 3.0.6, PHP 4]

[utf8_decode\(\)](#) décode la chaîne *data*, en supposant qu'elle est au format UTF-8, et la convertit au format ISO-8859-1.

Voir aussi [utf8_encode\(\)](#) pour plus de détails sur le codage UTF-8.

[10.76.23 utf8_encode](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [utf8_encode](#)(string *data*)
[PHP 3>= 3.0.6, PHP 4]

[utf8_encode\(\)](#) code la chaîne *data* au format UTF-8, et retourne la version codée. UTF-8 est un mécanisme standardisé utilisé par Unicode pour coder les caractère de grande taille dans des flots d'octets. UTF-8 est

transparent pour les caractères ASCII, il est auto-synchronisé (c'est à dire qu'un programme peut toujours savoir dans un flot d'octet où un caractère commence), et peut être utilisé pour faire des comparaisons de chaînes standard, comme pour le tri. PHP utilise l'UTF-8 pour coder les caractères jusqu'à 4 octets comme ceci :

octets	bits	représentation
1	7	0bbbbbbb
2	11	110bbbbb 10bbbbbb
3	16	1110bbbb 10bbbbbb 10bbbbbb
4	21	11110bbb 10bbbbbb 10bbbbbb 10bbbbbb

Chaque *b* représente un bit qui peut être utilisé pour enregistrer un caractère.

[10.77 XSLT](#)

[\[Notes en ligne\]](#)

[10.77.1 Introduction](#)

[\[Notes en ligne\]](#)

[10.77.1.1 A propos de XSLT et Sablotron](#)

[\[Notes en ligne\]](#)

XSLT (Extensible Stylesheet Language (XSL) Transformations) est un langage de transformation des documents XML en d'autres documents XML. C'est un standard défini par le consortium World Wide Web (W3C). Les informations sur le XSLT et ses technologies sont disponibles à <http://www.w3.org/TR/xslt>.

[10.77.1.2 Installation](#)

[\[Notes en ligne\]](#)

Cette extension utilise *Sablotron* et *expat*, qui sont toutes les deux disponibles à <http://www.gingerall.com/>. Les sources comme les exécutables sont proposés.

Sous UNIX, lancez `configure` avec l'option `--with-sablot`. La librairie *Sablotron* doit être installée là où le compilateur peut la trouver.

[10.77.1.3 A propos de Sablotron](#)

[\[Notes en ligne\]](#)

Cette extension PHP implémente le support de *Sablotron*, par Ginger Alliance. Cette librairie vous permet de transformer des documents XML en d'autres documents XML, mais aussi en HTML ou encore n'importe quel format à balise. Elle fournit un mécanisme basique et portable de templates, séparant le contenu de l'interface d'un site web.

[10.77.2 xslt_closelog](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [xslt_closelog](#)(resource *xh*)

[PHP 4 >= 4.0.3]

xh

- Une référence valide sur un analyseur XSLT.

[`xslt_closelog\(\)`](#) retourne FALSE si ***xh*** n'est pas un analyseur XSLT valide, ou bien si la fermeture du fichier d'historique a échoué. Sinon, retourne TRUE.

10.77.3 xslt_create

[\[Notes en ligne\]](#) [\[Exemples\]](#)

resource [`xslt_create`](#)

[PHP 4 >= 4.0.3]

[`xslt_create\(\)`](#) retourne un identifiant d'analyseur XSLT. Il sera nécessaire pour les appels ultérieurs aux fonctions.

10.77.4 xslt_errno

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [`xslt_errno`](#)(*int xh*)

[PHP 4 >= 4.0.3]

[`xslt_errno\(\)`](#) le numéro courant d'erreur, pour l'analyseur ***xh***. Si ***xh*** n'est pas fourni, le dernier numéro d'erreur est retourné.

10.77.5 xslt_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

mixed [`xslt_error`](#)(*int xh*)

[PHP 4 >= 4.0.3]

[`xslt_error\(\)`](#) le message d'erreur courant, pour l'analyseur ***xh***. Si ***xh*** n'est pas fourni, le dernier numéro d'erreur est retourné.

10.77.6 xslt_fetch_result

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [`xslt_fetch_result`](#)(*int xh string result_name*)

[PHP 4 >= 4.0.3]

[`xslt_fetch_result\(\)`](#) lit le résultat disponible dans le buffer de l'analyseur ***xh***. Si ***result_name*** n'est pas fourni, le résultat `"/_result"` sera lu.

[10.77.7 xslt_free](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [xslt_free](#)(*resource* *xh*)

[PHP 4 >= 4.0.3]

[xslt_free\(\)](#) détruit l'analyseur XSLT *resource* *xh*.

[10.77.8 xslt_openlog](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [xslt_openlog](#)(*resource* *xh* *string* *logfile* *int* *loglevel*)

[PHP 4 >= 4.0.3]

[xslt_openlog\(\)](#) remplace le fichier d'historique courant par *logfile*, pour l'analyseur XSLT *xh*.

[10.77.9 xslt_output_begintransform](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [xslt_output_begintransform](#)(*string* *xslt_filename*)

[PHP 4 >= 4.0.3]

[xslt_output_begintransform\(\)](#) commence à retourner la transformée de vos données. En l'appel de [xslt_output_begintransform\(\)](#) et celui de [xslt_output_endtransform\(\)](#), toutes les données seront transformées par la feuille de style XSLT *string* *xslt_filename*.

Transformation avec une feuille de style XSLT (avec DOM-XML)

```
<?php
$xml_file = "article.xml";
xslt_output_begintransform($xml_file);
$doc = new_xmldoc('1.0');
$article = $doc->new_root('article');
$article->new_child('title', 'Histoire du Tyrol méridional');
$article->new_child('author', 'Sterling Hughes');
$article->new_child('body', 'Juste après la première guerre mondiale, l'Italie
obtient le Tyrol méridional aux dépends de l'Autriche.
Et depuis cette époque, rien d'interessant n'est arrivé.');
```

[10.77.10 xslt_output_endtransform](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

void [xslt_output_endtransform](#)

[PHP 4 >= 4.0.3]

[xslt_output_endtransform\(\)](#) termine la transformation commencée avec [xslt_output_begintransform\(\)](#). Vous

devez appeler cette fonction pour pouvoir accéder aux résultats.

10.77.11 xslt_process

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [xslt_process\(\)](#) (*string xsl_data string xml_data string result*)

[PHP 4 >= 4.0.3]

[xslt_process\(\)](#) prend la chaîne *string xsl_data* comme feuille de style XSLT, et des données XML dans *xml_data*. Le résultat de la transformation sera placé dans *result*. [xslt_process\(\)](#) retourne TRUE en cas de succès, et FALSE sinon. Vous pourrez lire les erreurs survenues grâce aux fonctions [xslt_errno\(\)](#) et [xslt_error\(\)](#) functions.

Utilisation de xslt_process() pour transformer trois

```
<?php
$xmlData = '
<xsl:stylesheet
version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="article">
    <table border="1" cellpadding="2" cellspacing="1">
        <tr>
            <td width="20%">

                </title>
                <td width="80%">
                    <h2><xsl:value-of select="title"></h2>
                    <h3><xsl:value-of select="author"></h3>
                    <br>
                    <xsl:value-of select="body">
                </td>
            </tr>
        </table>
    </xsl:template>
</xsl:stylesheet>';
$xmlData = '
<?xml version="1.0">
<article>
    <title>Learning German</title>
    <author>Sterling Hughes</author>
    <body>
Essential phrases:
    <br>
    <br>
Komme sie mir sagen, woe die toilette es?<br>
Eine grande beer bitte!<br>
Noch einem bitte.<br>
    </body>
</article>';
if (xslt_process($xmlData, $xmlData, $result))
{
    echo "Voici un brillant article sur l'apprentissage du ";
    echo " français: ";
    echo "<br>\n<br>";
    echo $result;
}
else
```

```
{
echo "Une erreur est survenue durant le traitement XSL...\n";
echo "\tErreur numéro : " . xslt_errno() . "\n";
echo "\tMessage d'erreur : " . xslt_error() . "\n";
exit;
}
?>
```

10.77.12 xslt_run

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [xslt_run](#)(*resource \$xh string \$xslt_file string \$xml_data_file string \$result array \$xslt_params array \$xslt_args*)

[PHP 4 >= 4.0.3]

[xslt_run\(\)](#) applique e au fichier *\$xml_data_file* la feuille de style *\$string \$xslt_file*. La feuille de style peut utiliser les paramètres optionnels *\$xslt_params* et l'analyseur XSLT est démarré avec *array \$xslt_args*. Le résultat est placé dans le buffer nommé *\$result* (par défaut, dans *"/_result"*).

10.77.13 xslt_set_sax_handler

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [xslt_set_sax_handler](#)(*resource \$xh array \$handlers*)

[PHP 4 >= 4.0.3]

[xslt_set_sax_handler\(\)](#) remplace les gestionnaires SAX de l'analyseur XSLT *\$xh* par *\$handlers*.

10.77.14 xslt_transform

[\[Notes en ligne\]](#) [\[Exemples\]](#)

bool [xslt_transform](#)(*string \$xsl string \$xml string \$result string \$params string \$args string \$resultBuffer*)

[PHP 4 >= 4.0.3]

[xslt_transform\(\)](#) fourni une interface avec les API avancées de sablotron, sans vous obliger à utiliser les ressources.

10.78 YAZ

[\[Notes en ligne\]](#)

10.78.1 Introduction

[\[Notes en ligne\]](#)

Cette extension offre à PHP l'interface avec les produits **YAZ**, qui implémentent le protocole Z39.50. Avec cette extension, vous pouvez facilement implémenter un client Z39.50 qui fouille des serveurs Z39.50 en parallèle.

YAZ est disponible à <http://www.indexdata.dk/yaz/>. Vous pouvez trouver des informations, des scripts d'exemples, etc... pour cette extension à <http://www.indexdata.dk/phpyaz/>. Le module masque l'essentiel de la complexité de Z39.50, ce qui le rend très facile à utiliser. Il supporte les connexions persistantes de manière similaire à celle supportées par les serveurs SQL : cela signifie qu'une connexion est partagée entre plusieurs scripts PHP, ce qui évite les opérations de connexions.

10.78.2 Installation

[\[Notes en ligne\]](#)

Compilez YAZ et installez le. Compilez PHP avec vos modules et ajoutez l'option `--with-yaz`. Les instructions sont :

```
gunzip -c yaz-1.6.tar.gz|tar xf -
gunzip -c php-4.0.X.tar.gz|tar xf -
cd yaz-1.6
./configure --prefix=/usr
make
make install
cd ../php-4.0.X
./configure --with-yaz=/usr/bin
make
make install
```

10.78.3 Exemple

[\[Notes en ligne\]](#)

PHP/YAZ conserve les connexions aux serveurs. Un entier positif représente l'ID d'une connexion particulière.

Le script ci-dessous montre comment effectuer une recherche parallèle. Lorsqu'il est appelé sans paramètre, ce script affiche la requête. Sinon, il effectue la recherche sur les serveurs.

YAZ

```
<?php
$num_hosts = count ($host);
if (empty($term) || count($host) == 0) {
echo '<form method="get">
    <input type="checkbox"
name="host[]" value="bagel.indexdata.dk/gils">
GILS test
    <input type="checkbox"
name="host[]" value="localhost:9999/Default">
local test
    <input type="checkbox" checked="1"
name="host[]" value="z3950.bell-labs.com/books">
BELL Labs Library
    <br>
RPN Query:
    <input type="text" size="30" name="term">
    <input type="submit" name="action" value="Search">
    ';
} else {
echo 'Vous avez recherché ' . htmlspecialchars($term) . '<br>';
```

```

for ($i = 0; $i < $num_hosts; $i++) {
    $id[] = yaz_connect($host[$i]);
    yaz_syntax($id[$i], "sutrs");
    yaz_search($id[$i], "rpn", $term);
}
yaz_wait();
for ($i = 0; $i < $num_hosts; $i++) {
    echo '<hr>' . $host[$i] . " ";
    $error = yaz_error($id[$i]);
    if (!empty($error)) {
        echo "Erreur: $error";
    } else {
        $hits = yaz_hits($id[$i]);
        echo "Nombre de résultats : $hits";
    }
    echo '<dl>';
    for ($p = 1; $p <= 10; $p++) {
        $rec = yaz_record($id[$i], $p, "string");
        if (empty($rec)) continue;
        echo "<dt><B>$p</B></dt><dd>";
        echo ereg_replace("\n", "<br>\n", $rec);
        echo "</dd>";
    }
    echo '</dl>';
}
?>

```

10.78.4 yaz_addinfo

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_addinfo](#) (int *id*)
 [PHP 4 >= 4.0.1]

[yaz_addinfo\(\)](#) retourne plus de détails après la dernière erreur survenue. Une chaîne vide est retournée si la dernière opération a été réussie, ou bien si aucune autre information n'est disponible.

10.78.5 yaz_close

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_close](#) (int *id*)
 [PHP 4 >= 4.0.1]

[yaz_close\(\)](#) ferme une connexion à un hôte YAZ. L'application ne peut plus utiliser l'identifiant de connexion *id*.

10.78.6 yaz_connect

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_connect](#) (string *zurl*)
 [PHP 4 >= 4.0.1]

[yaz_connect\(\)](#) prépare une connexion à un serveur Z39.50 target. *zurl* est de la forme host[:port][/database]. Si port est omis, 210 est utilisé. Si database est omis, Default est utilisé. Cette fonction n'est pas bloquante, et ne tente pas d'établir une socket. En fait, elle ne fait que préparer la connexion pour exécution ultérieure par [yaz_wait\(\)](#).

10.78.7 yaz_errno

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_errno](#) (int *id*)
[PHP 4 >= 4.0.1]

[yaz_errno\(\)](#) retourne le numéro d'erreur de la dernière requête. Une valeur positive est retournée si le serveur a retournée un diagnostic. La valeur 0 est retournée si aucune erreur n'est survenue. Une valeur négative indique une erreur sans diagnostic.

[yaz_errno\(\)](#) doit être appelée après chaque requête. (après la fin de [yaz_wait\(\)](#)), pour savoir si la transaction a réussi ou échoué.

10.78.8 yaz_error

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_error](#) (int *id*)
[PHP 4 >= 4.0.1]

[yaz_error\(\)](#) retourne un message d'erreur pour la dernière requête. Une chaîne vide est retournée si la dernière requête a réussi.

[yaz_error\(\)](#) retourne un message en anglais, qui correspond au numéro d'erreur retourné par [yaz_errno\(\)](#).

10.78.9 yaz_hits

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_hits](#) (int *id*)
[PHP 4 >= 4.0.1]

[yaz_hits\(\)](#) retourne le nombre de résultat de la dernière recherche.

10.78.10 yaz_range

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_range](#) (int *id*, int *start*, int *number*)
[PHP 4 >= 4.0.1]

[yaz_range\(\)](#) est utilisée conjointement à [yaz_search\(\)](#), pour spécifier le nombre maximal *number* de résultat à lire, ainsi que la position de début de lecture avec *start*. Si [yaz_range\(\)](#) n'est pas utilisée, *start* vaudra 1 et *number* vaudra 10.

Retourne TRUE en cas de succès; FALSE en cas d'erreur.

10.78.11 [yaz_record](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_record](#)(int *id*, int *pos*, string *type*)

[PHP 4 >= 4.0.1]

[yaz_record\(\)](#) retourne un résultat à la position *pos*, ou une chaîne vide si aucun résultat n'est disponible à la position *pos*.

[yaz_record\(\)](#) recherche une ligne dans le résultat, à la position spécifiée. Si aucune ligne n'existe à la position donnée, une chaîne vide est retournée. L'argument *type* spécifie la forme du résultat retourné : si *type* vaut "string", la ligne est retournée sous la forme d'une chaîne prête à l'affichage. Si *type* vaut "array", la ligne sera retournée sous la forme d'un tableau structuré.

10.78.12 [yaz_search](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_search](#)(int *id*, string *type*, string *query*)

[PHP 4 >= 4.0.1]

[yaz_search\(\)](#) prépare une recherche sur le serveur identifié par *id*. *type* représente le type de requête : seul RPN est supporté actuellement, et dans ce cas, le troisième argument est un préfixe de notation de requête utilisé par YAZ. Comme pour [yaz_connect\(\)](#), [yaz_search\(\)](#) n'est pas bloquante, et ne fait que préparer la recherche pour exécution ultérieure, avec [yaz_wait\(\)](#).

Les requêtes RPN sont des représentation textuelles des requêtes de type Type-1, comme définit dans le standard Z39.50. Cependant, dans la représentation textuelle utilisée par YAZ, une notation à préfixage est utilisée, c'est à dire que l'opérateur précède l'opérande. La chaîne de requête est une séquence de mots réservés, où les espaces sont ignorés, à moins qu'ils n'aient été mis entre guillemets doubles. Les mots réservés qui commencent par un arobase (@) sont considérés comme des opérateurs et traités comme tels.

Syntaxe	Description
@and query1 query2	intersection des requêtes query1 et query2
@or query1 query2	union des requêtes query1 et query2
@not query1 query2	requêtes "query1 et non(query2)"
nomme le résultat	
@attrset set query	spécifie le jeu d'attributs de la requête. Cette construction n'est autorisée qu'une seule fois, au début d'une requête.
@attr set type=valeur query	Applique les attributs à une requête. Le type et la valeur sont des entiers indiquant les types et valeurs des attributs, dans cet ordre. Le jeu, si fourni, spécifie le jeu d'attribut utilisé. @tab

Les requêtes suivantes illustrent des requêtes valides :

```
ordinateur
```

Recherche les documents qui contiennent le mot "ordinateur". Aucun attribut n'est spécifié.

```
"serveur rapide"
```

Recherche les documents qui contiennent les mots "serveur rapide"

```
@attr 1=4 php
```

L'attribut est de type 1 (Bib=1 use), sa valeur est 4 (Title, titre) : cette requête recherche les documents où le mot "php" est dans le titre.

```
@attrset gils @and @attr 1=4 php @attr 1=1003 "Rasmus Lerdorf"
```

Cette requête utilise tout le jeu d'attributs GILS. Elle recherche les documents dont le titre contient "php", et qui contiennent le nom "Rasmus Lerdorf" comme auteur.

[10.78.13 yaz_syntax](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_syntax](#)(int *id*, string *syntax*)

[PHP 4 >= 4.0.1]

[yaz_syntax\(\)](#) est utilisée conjointement avec [yaz_search\(\)](#) pour spécifier la méthode de lecture des lignes.

[10.78.14 yaz_wait](#)

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [yaz_wait](#)(int *id*, string *syntax*)

[PHP 4 >= 4.0.1]

[yaz_wait\(\)](#) exécute les requêtes préparée par les fonctions [yaz_connect\(\)](#), [yaz_search\(\)](#), [yaz_wait\(\)](#) fonctionne en mode bloquant. [yaz_wait\(\)](#) se termine lorsque toutes les requêtes se sont terminées (succès ou erreur).

[10.79 Zlib \(Compression\)](#)

[\[Notes en ligne\]](#)

Ce module utilise les fonctions de la librairie zlib ([zlib](#)) de Jean-loup Gailly et Mark Adler pour lire et écrire, de manière transparente, des fichiers compressés avec gzip (.gz). Il faut utiliser la librairie zlib, de version >= 1.0.9.

Ce module contient des versions de la plus part des fonctions du chapitre [10.22 Système de fichiers](#). Mais celles-ci fonctionnent non seulement avec des fichiers compressés, mais aussi des fichiers décompressés (hormis les fonctions utilisant les sockets).

[10.79.1 Petit exemple](#)

[\[Notes en ligne\]](#)

Ouvre un fichier temporaire, écrit un texte et puis affiche deux fois le contenu.

Petit exemple

```
<?php
    $filename = tempnam('/tmp', 'zlibtest').'.gz';
print "<html>\n<head></head>\n<body>\n<pre>\n";
    $s = "Only a test, test, test, test, test, test, test, test!\n";
    // ouvre un fichier en écriture, avec compression maximale
    $zp = gzopen($filename, "w9");
    // écrit la chaîne dans le fichier
gzwrite($zp, $s);
    // ferme le fichier
gzclose($zp);
    // ouvre en lecture
    $zp = gzopen($filename, "r");
    // lis 3 caractères
print gzread($zp, 3);
    // Affiche le reste du fichier
gzpassthru($zp);
print "\n";
    // ouvre le fichier, et affiche le contenu (deuxième passe)
if (readgzfile($filename) != strlen($s)) {
echo "Error with zlib functions!";
}
unlink($filename);
print "<pre>\n</hl></body>\n</html>\n";
?>
```

10.79.2 gzclose

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzclose](#)(int *zp*)

[PHP 3, PHP 4]

[gzclose\(\)](#) ferme le pointeur *zp*.

[gzclose\(\)](#) retourne TRUE ou FALSE, suivant le succès ou l'échec.

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

10.79.3 gzeof

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzeof](#)(int *zp*)

[PHP 3, PHP 4]

Retourne TRUE si le pointeur interne du fichier compressé est à la fin du fichier, sinon retourne FALSE.

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

10.79.4 gzfile

[\[Notes en ligne\]](#) [\[Exemples\]](#)

array [gzfile](#)(string *filename*, int *use_include_path*)

[PHP 3, PHP 4]

Identique à [readgzfile\(\)](#), mais [gzfile\(\)](#) retourne un tableau.

Vous pouvez aussi utiliser le deuxième paramètre optionnel en le mettant à "1", si vous voulez rechercher le fichier dans le dossier [7.1.1.14 ini.include-path](#).

Voir aussi [readgzfile\(\)](#), et [gzopen\(\)](#).

10.79.5 gzgetc

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzgetc](#)(int *zp*)

[PHP 3, PHP 4]

Retourne une chaîne décompressée, qui contient un caractère unique, lu depuis le fichier référencé par le pointeur *zp*. Retourne FALSE à la fin du fichier (voir [gzeof\(\)](#)).

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

Voir aussi [gzopen\(\)](#), et [gzgets\(\)](#).

10.79.6 gzgets

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzgets](#)(int *zp*, int *length*)

[PHP 3, PHP 4]

[gzgets\(\)](#) retourne une chaîne décompressée, de longueur inférieure ou égale à *length* – 1 octets, lue depuis le fichier référencé par le pointeur de fichier *zp*. La lecture s'interrompt lorsque *length* – 1 octets ont été lus, ou bien lorsqu'une nouvelle ligne a été rencontrée, ou bien lorsque la fin du fichier a été atteinte.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier compressé doit être valide, et doit référencer un fichier qui a été ouvert par [gzopen\(\)](#).

Voir aussi [gzopen\(\)](#), [gzgetc\(\)](#), et [fgets\(\)](#).

10.79.7 gzgetss

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzgetss](#)(int *zp*, int *length*, string *allowable_tags*)

[PHP 3, PHP 4]

[gzgetss\(\)](#) est identique à [gzgets\(\)](#), mais elle essaie de supprimer toutes les balises HTML et PHP du texte lu.

Vous pouvez utiliser le troisième argument optionnel pour indiquer les balises qui ne doivent pas être supprimées. Note : *allowable_tags* a été ajouté en PHP 3.0.13, PHP 4.0B3.

Voir aussi [gzgets\(\)](#), [gzopen\(\)](#), et [strip_tags\(\)](#).

10.79.8 gzopen

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzopen](#)(string *filename*, string *mode*, int *use_include_path*)

[PHP 3, PHP 4]

[gzopen\(\)](#) ouvre un fichier compressé avec gzip (.gz) pour le lire ou l'écrire. Le paramètre de mode est le

même que dans [fopen\(\)](#) ("rb" ou "wb") mais il peut aussi inclure un niveau de compression ("wb9") ou une stratégie: 'f' pour les données filtrées, comme dans "wb6f", 'h' pour Huffman seul, comme dans "wb1h". (Voir la description de deflateInit2 dans zlib.h pour plus de détails à propos des paramètres de stratégie).

[gzopen\(\)](#) peut être utilisé pour ouvrir des fichiers qui ne sont pas au format gzip; dans ce cas, [gzread\(\)](#) lira directement le fichier, sans appliquer de décompression.

[gzopen\(\)](#) retourne un pointeur de fichier sur le fichier ouvert. Ce pointeur sera nécessaire pour toutes les opérations ultérieures sur ce fichier. Les opérations de compression/décompression seront transparentes. Si l'ouverture échoue, la fonction retourne FALSE.

Vous pouvez utiliser le paramètre optionnel en le mettant à "1", si vous voulez rechercher le fichier dans le dossier [7.1.1.14 ini.include-path](#).

Exemple gzopen()

```
<?php
$fp = gzopen("/tmp/file.gz", "r");
?>
```

Voir aussi [gzclose\(\)](#).

10.79.9 gzpassthru

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzpassthru](#)(int *zp*)

[PHP 3, PHP 4]

Lit toutes les informations d'un fichier compressé jusqu'à EOF et écrit le résultat décompressé dans la sortie standard.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et avoir été ouvert par [gzopen\(\)](#).

Le fichier pointé est refermé par [gzpassthru\(\)](#) ce qui le rend inutilisable pour les opérations ultérieures.

10.79.10 gzputs

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzputs](#)(int *zp*, string *str*, int *length*)

[PHP 3, PHP 4]

[gzputs\(\)](#) est un alias de [gzwrite\(\)](#), et lui est identique en tous points.

10.79.11 gzread

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzread](#)(int *zp*, int *length*)

[PHP 3, PHP 4]

[gzread\(\)](#) lit jusqu'à *length* octets depuis le fichier compressé référencé par *zp*. La lecture stoppe lorsque *length* octets décompressés ont été lus, ou que la fin du fichier a été trouvée.

```
<?php
// lis le contenu d'un fichier compressé et le met dans une chaîne
$filename = "/usr/local/something.txt.gz";
$zd = gzopen( $filename, "r" );
$contents = gzread( $zd, 10000 );
gzclose( $zd );
?>
```

Voir aussi [gzwrite\(\)](#), [gzopen\(\)](#), [gzgets\(\)](#), [gzgetss\(\)](#), [gzfile\(\)](#), et [gzpassthru\(\)](#).

10.79.12 gzrewind

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzrewind](#)(int *zp*)

[PHP 3, PHP 4]

[gzrewind\(\)](#) positionne le pointeur interne du fichier au début du fichier compressé.

Si une erreur survient, retourne 0.

Le pointeur de fichier doit être valide, et avoir été retourné par la fonction [gzopen\(\)](#).

Voir aussi [gzseek\(\)](#) et [gztell\(\)](#).

10.79.13 gzseek

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzseek](#)(int *zp*, int *offset*)

[PHP 3, PHP 4]

[gzseek\(\)](#) positionne le pointeur interne du fichier compressé *zp* à la position donnée en *offset*. Equivalent à l'appel (en C) de `gzseek(zp, offset, SEEK_SET)`.

Si le fichier est ouvert en lecture seule, cette fonction émulée peut être extrêmement lente. Si le fichier est ouvert en lecture, seul le déplacement avant (forward seek) sont acceptés. `gzseek` compresse alors une séquence de zéro jusqu'à la nouvelle position.

[gzseek\(\)](#) retourne 0 en cas de succès, sinon retourne -1. Notez que positionner le pointeur au delà de la fin du fichier est une erreur.

Voir aussi [gztell\(\)](#) et [gzrewind\(\)](#).

10.79.14 gztell

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gztell](#)(int *zp*)

[PHP 3, PHP 4]

[gztell\(\)](#) retourne la position du pointeur interne du fichier référencé par *zp*, i.e., son offset en octets depuis le début du fichier.

Si une erreur survient, retourne FALSE.

Le pointeur de fichier doit être valide, et doit avoir été retourné par la fonction [gzopen\(\)](#).

Voir aussi [gzopen\(\)](#), [gzseek\(\)](#) et [gzrewind\(\)](#).

10.79.15 gzwrite

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [gzwrite](#)(int *zp*, string *string*, int *length*)

[PHP 3, PHP 4]

[gzwrite\(\)](#) écrit le contenu de la chaîne *string* dans le fichier compressé référencé par le pointeur *zp*. Si l'argument *length* est donné, l'écriture cessera après avoir écrit *length* octets (non compressé), ou bien si la fin de la chaîne a été atteinte.

Notez que si l'argument *length* est donnée, alors l'option [7.1.1.18 ini.magic-quotes-runtime](#) sera ignorée et les slashes ne seront pas supprimés de la chaîne *string*.

Voir aussi [gzread\(\)](#), [gzopen\(\)](#), et [gzputs\(\)](#).

10.79.16 readgzfile

[\[Notes en ligne\]](#) [\[Exemples\]](#)

int [readgzfile](#)(string *filename*, int *use_include_path*)

[PHP 3, PHP 4]

[readgzfile\(\)](#) lit un fichier, le décompresse et l'écrit dans la sortie standard.

[readgzfile\(\)](#) peut être utilisé pour lire un fichier qui n'est pas au format gzip, car dans ce cas, la décompression est omise, et le fichier est directement affiché.

[readgzfile\(\)](#) retourne le nombre d'octets (non compressé) lus depuis le fichier. Si une erreur survient, retourne FALSE, et à moins que la fonction n'ait été appelée avec [@readgzfile](#), l'erreur est affichée.

Le fichier *filename* sera ouvert et son contenu sera écrit dans la sortie standard.

Vous pouvez utiliser le paramètre optionnel en le mettant à "1", si vous voulez rechercher le fichier dans le dossier [7.1.1.14 ini.include-path](#).

Voir aussi [gzpassthru\(\)](#), [gzfile\(\)](#), et [gzopen\(\)](#).

10.79.17 gzcompress

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzcompress](#)(string *data*, int *level*)

[PHP 4 >= 4.0.1]

[gzcompress\(\)](#) retourne la version compressée GZ de la chaîne *data*, ou FALSE en cas d'erreur. Le paramètre *level* peut prendre des valeurs depuis 0 (pas de compression) jusqu'à 9 (compression maximum).

Voir aussi [gzuncompress\(\)](#).

10.79.18 gzuncompress

[\[Notes en ligne\]](#) [\[Exemples\]](#)

string [gzuncompress](#)(string *data*, int *length*)

[PHP 4 >= 4.0.1]

[gzuncompress\(\)](#) prend la chaîne *data* en entrée (compressée par [gzcompress\(\)](#)) et retourne la chaîne originale, ou bien FALSE en cas d'erreur. [gzuncompress\(\)](#) retournera une erreur si la taille de la chaîne décompressée est plus de 256 fois la longueur de la chaîne compressée *data* ou plus que le paramètre optionnel *length*.

Note : Ceci n'est pas équivalent à une compression gzip, qui inclus en plus des données d'entête. Voir `@xref{function.gzencode , , gzencode() }` pour la compression gzip.
} Voir aussi [gzcompress\(\)](#).

A Débugueur PHP

[\[Notes en ligne\]](#)

A.1 A propos du debugueur

[\[Notes en ligne\]](#)

PHP 3 inclus le support d'un serveur de debugage.
PHP 4 n'inclus aucun support de debugage.

A.2 Utiliser le débugeur PHP

[\[Notes en ligne\]](#)

Le débugeur PHP sert à repérer les bugs récalcitrants. Le débugeur fonctionne en se connectant à un port **TCP** à chaque démarrage de PHP. Tous les messages d'erreur seront envoyés sur cette connexion. Cette page est faite pour un "serveur de débugeage", qui peut peut fonctionner avec un **IDE** ou un éditeur programmable (tel que Emacs).

Comment paramétrer le débugeur :

1. Réservez un port TCP pour le débugeur dans le fichier [7.1 Le fichier de configuration](#) ([7.1.4.2 ini.debugger.port](#)) et activez le ([7.1.4.3 ini.debugger.enabled](#)).
2. Configurer un client TCP sur ce port (par exemple `socket -l -s 1400` sous UNIX).
3. Dans votre code, placez la ligne "`debugger_on(host)`", où *host* est l'IP ou le nom de l'hôte qui supporte le client **TCP**.

Désormais, toutes les alertes, notes, ... seront envoyées sur la socket client, *même si vous avez inactivé le rapport d'erreur avec [error_reporting\(\)](#)*.

A.3 Protocole du débugeur

[\[Notes en ligne\]](#)

Le protocole de débugeage est ligne par ligne. Chaque ligne a un type *type* et plusieurs lignes composent un message. Chaque message commence avec une ligne du type *start* et se termine avec une ligne de type *end*. PHP peut envoyer des lignes de plusieurs messages simultanément.

Voici un exemple de ligne à ce format :

```
date time
host(pid)
type:
message-data
```

date

- Les dates sont au format ISO 8601 (*yyyy-mm-dd*)
time
- Les heures, y compris les micro-secondes: *hh:mm:uuuuuuu*
host
- Le nom DNS ou adresse IP de l'hôte qui a généré l'erreur.

pid

- PID (process id) sur l'hôte *host*, qui a généré l'erreur.

type

- Type de la ligne. Indique au programme client comment traiter les données suivantes :

Nom	Signification
start	Indique au programme client que le message du débogueur commence ici. Le contenu de <i>data</i> sera un type d'erreur, comme listé ci dessous. @tab
message	Le message d'erreur PHP.
location	Nom du fichier et numéro de ligne, où l'erreur est survenue. La première occurrence de location contiendra toujours la localisation générale. <i>data</i> contiendra : <i>file:line</i> . Il y a toujours une indication de location après un message et après chaque fonction. @tab
frames	Nombre de frames dans le dump de la pile. Si il y a 4 frames, attendez vous à des information sur 4 niveaux de fonctions. Si la ligne "frame" n'existe pas, la profondeur doit être 0 (une erreur est survenue au niveau général). @tab
function	Nom de la fonction qui a généré l'erreur. Elle sera répétée à chaque niveau de la pile d'appel. @tab
end	Indique au client que le message du débogueur se termine ici. @tab

data

- Ligne de données.

Débogueur	Interne PHP
alerte (warning)	E_WARNING
erreur	E_ERROR
analyse (parse)	E_PARSE
note (notice)	E_NOTICE
core-error	E_CORE_ERROR
core-warning	E_CORE_WARNING
inconnue	(toutes les autres)

Exemple de message du débogueur

```
1998-04-05 23:27:400966 lucifer.guardian.no(20481) start: notice
1998-04-05 23:27:400966 lucifer.guardian.no(20481) message: Uninitialized variable
1998-04-05 23:27:400966 lucifer.guardian.no(20481) location: (null):7
1998-04-05 23:27:400966 lucifer.guardian.no(20481) frames: 1
1998-04-05 23:27:400966 lucifer.guardian.no(20481) function: display
1998-04-05 23:27:400966 lucifer.guardian.no(20481) location: /home/ssb/public_html/test.php3:10
1998-04-05 23:27:400966 lucifer.guardian.no(20481) end: notice
```

B Migration de PHP/FI 2.0 à PHP 3.0

[\[Notes en ligne\]](#)

B.1 A propos des incompatibilités en 3.0

[\[Notes en ligne\]](#)

PHP 3.0 a été entièrement réécrit. Le nouvel analyseur syntaxique est beaucoup plus robuste et cohérent qu'en version 2.0. Il est aussi nettement plus rapide et utilise encore moins de mémoire. Cependant, ces améliorations n'ont pu être possible qu'au prix de modifications parfois importantes, tant au niveau des syntaxes, qu'au niveau des fonctionnalités.

De plus, l'équipe de développement PHP a essayé de nettoyer la syntaxe et les sémantiques, ce qui a aussi causé quelques incompatibilités. A long terme, nous pensons que ces modifications seront pour le bien de tous.

Ce chapitre va tenter de vous montrer les incompatibilités que vous pourriez rencontrer lors de votre migration de PHP/FI 2.0 à PHP 3.0 et de vous aider à les résoudre. Les nouvelles fonctionnalités ne sont pas signalées, à moins que cela ne soit nécessaire.

Un programme de conversion automatique de vos vieux script PHP/FI 2.0 existe. Il est disponible dans le dossier de convertisseur de la distribution PHP 3.0. Ce programme ne fait que repérer les modifications de syntaxe et ne vous épargnera pas une relecture attentive du script.

B.2 Balises PHP

[\[Notes en ligne\]](#)

La première chose que vous remarquerez probablement est que les balises de PHP start et end ont changé. L'ancienne forme `<? ?>` a été remplacée par trois nouvelles balises possibles :

Migration: Migration: balises start/end

```
<?php
echo "Ceci est du code PHP/FI 2.0.\n";?
?>
```

Comme en version 2.0, PHP/FI accepte aussi cette variante :

Migration: premières nouvelles balises PHP

```
<?php
echo "Ceci est du code PHP 3.0!\n";
?>
```

Notez bien que la balise de fin contient désormais un point d'interrogation et un signe supérieur `">"`. Cependant, si vous souhaitez utiliser XML sur votre serveur, vous aurez sûrement des problèmes avec cette variante, car PHP risque d'essayer d'exécuter des balises XML. A cause de ceci, la notation suivante a été ajoutée :

Migration: Nouvelles balises PHP

```
<?php
echo "Ceci est du code PHP 3.0!\n";
?>
```

Certains d'entre vous rencontrent des problèmes avec les éditeurs qui ne comprennent pas ce type de balises

d'instruction : Microsoft FrontPage est l'un de ces éditeurs, et, pour contourner le problème, la variation suivante a été introduite :

Nouvelles balises PHP

```
<script language="php">
echo "Ceci est du code PHP 3.0!\n";
</script>
```

B.3 if..endif syntax

[\[Notes en ligne\]](#)

La syntaxe alternative pour écrire des instructions if/elseif/else, avec if(); elseif(); else; endif; ne pouvait pas être conservée sans ajouter beaucoup de complexité à l'analyseur syntaxique. De ce fait, cette syntaxe a changée :

Migration: ancienne syntaxe if..endif

```
<?php
if ($foo);
echo "oui\n";
elseif ($bar);
echo "presque\n";
else;
echo "non\n";
endif;
?>
```

Migration: nouvelle syntaxe if..endif

```
<?php
if ($foo):
echo "oui\n";
elseif ($bar):
echo "presque\n";
else:
echo "non\n";
endif;
?>
```

Notez que les points virgules ont été remplacée par des points dans toutes les commandes, sauf pour la dernière expression (endif).

B.4 while syntax

[\[Notes en ligne\]](#)

Tout comme pour if..endif, la syntaxe des boucles while..endwhile a changée :

Migration: ancienne syntaxe while..endwhile

```
<?php
while ($more_to_come);
...
```

```
endwhile;
?>
```

Migration: nouvelle syntaxe while..endwhile

```
<?php
while ($more_to_come):
    ...
endwhile;
?>
```

Attention : si vous utilisez la vieille syntaxe while..endwhile en PHP 3.0, vous obtiendrez une boucle sans fin !

B.5 Types d'expression

[\[Notes en ligne\]](#)

PHP/FI 2.0 utilisait le membre à gauche dans les expressions, pour déterminer le type de résultat attendu. PHP 3.0 prend en compte les deux cotés de l'expression et cela peut produire des résultats inattendus avec les scripts 2.0.

Considérez les lignes suivantes:

```
<?php
$a[0]=5;
$a[1]=7;
$key = key($a);
while ("" != $key) {
    echo "$keyn";
    next($a);
}
?>
```

En PHP/FI 2.0, cet exemple va afficher les indices des \$a. En PHP 3.0, l'exemple ne va rien afficher du tout. La raison est qu'en PHP 2.0, puisque l'argument de gauche est de type chaîne, une comparaison de chaîne était effectuée et, effectivement, "" n'est pas "", ce qui conduit la boucle à continuer. En PHP 3, lorsqu'une chaîne est comparée avec un entier, la comparaison est de type chaîne (la chaîne est convertie en entier). Ce qui revient à faire la comparaison entre (atoi("")) qui vaut 0 et la variable qui vaut aussi 0 et comme 0==0, la boucle ne commence même pas.

La correction de ceci est simple : il suffit de remplacer les commandes while par:

```
<?php
while ((string)$key != "") {
    ...
}
?>
```

B.6 Les messages d'erreur ont changé

[\[Notes en ligne\]](#)

Les messages d'erreur en PHP 3.0 sont généralement plus précis que ceux de la version 2.0., mais vous ne

verrez plus la portion de code qui a causé l'erreur. A la place, un numéro de ligne et un nom de fichier sera retourné.

B.7 Evaluation rapide des booléens

[\[Notes en ligne\]](#)

En PHP 3., l'évaluation des est court-circuité. Cela signifie dans une expression telle que `((1 || test_me()))`, la fonction `test_me()` ne sera pas exécutée, car cela ne changera pas le résultat. C'est une amélioration mineure, mais qui peut avoir des effets secondaires importants.

B.8 La valeur TRUE/FALSE comme retour de fonctions

[\[Notes en ligne\]](#)

La plupart des fonctions internes de PHP ont été réécrite pour qu'elle retourne TRUE en cas de succès, et FALSE en cas d'erreur, au contraire des fonctions qui retournaient 0 et -1 en PHP/FI 2.0. Le nouveau comportement est beaucoup plus logique, comme par exemple `$fp = fopen("/your/file")` or `fail("fichier non trouvé!")`; Etant donné que PHP/FI 2.0 n'a pas de règle claire à propos de ce que les fonctions doivent retourner en cas d'échec, la plus part des scripts devront probablement être vérifié manuellement, après avoir utilisé le convertisseur 2.0 à 3.0.

Migration depuis 2.0: valeur retournées, ancienne façon

```
<?php
$fp = fopen($file, "r");
if ($fp == -1);
echo("Impossible d'ouvrir le fichier $file en lecture <br>\n");
endif;
?>
```

Migration depuis 2.0: valeur retournées, nouvelle façon

```
<?php
$fp = @fopen($file, "r") or print("Impossible d'ouvrir le fichier $file en lecture<br>\n");
?>
```

B.9 Diverses incompatibilités

[\[Notes en ligne\]](#)

- Le module PHP 3.0 pour Apache n'accepte plus les versions d'Apache antérieure à la version 1.2. Apache 1.2 ou plus récent est nécessaire.
- [echo\(\)](#) n'utilise plus de chaîne de formatage. Il faut utiliser [printf\(\)](#) à la place.
- En PHP/FI 2.0, un effet secondaire de l'implémentation faisait que `$foo[0]` était la même chose que `$foo`. Ce n'est plus vrai en PHP 3.0.
- Lire un tableau avec `$array[]` n'est plus valable. Ainsi, il n'est plus possible de passer en revue un tableau avec des boucles telles que `$data = $array[]`. Utilisez [current\(\)](#) et [next\(\)](#) à la place.

Ainsi, `$array1[] = $array2` n'ajoute pas les valeurs de `$array2` à `$array1`, mais crée un nouvel élément dans `$array1` et y affecte `$array2` comme dernier élément. Voir aussi les tableaux multidimensionnels.

- "+" n'est plus utilisable comme opérateur de concaténation de chaîne. A la place, il convertit les arguments en nombres et effectue une addition numérique. Utilisez "." à la place.

Migration depuis 2.0: concaténation de chaînes

```
<?php
echo "1" + "1";
?>
```

En PHP 2.0 cela retournerait 11, en PHP 3.0 cela va retourner 2. A la place, faites :

```
<?php
echo "1"."1";
?>
```

```
<?php
$a = 1;
$b = 1;
echo $a + $b;
?>
```

Cela va afficher 2, tant en PHP 2.0 qu'en 3.0.

```
<?php
$a = 1;
$b = 1;
echo $a.$b;
?>
```

Cela va afficher 11 en PHP 3.0.

C Développement PHP

[\[Notes en ligne\]](#)

C.1 Créer une fonction PHP 3

[\[Notes en ligne\]](#)

C.1.1 Prototypes de fonctions

[\[Notes en ligne\]](#)

Toutes les fonctions suivent le schéma suivant :

```
void php3_foo(INTERNAL_FUNCTION_PARAMETERS) {  
}
```

Même si votre fonction ne prend aucun argument, c'est comme cela qu'elle doit être appelée.

C.1.2 Arguments de fonctions

[\[Notes en ligne\]](#)

Les arguments sont toujours de type val. Ce type contient un membre de type union, qui indique le type réel d'argument. De cette façon, si votre fonction prend deux arguments, elle ressemble à ceci :

Argument de fonction de lecture

```
pval *arg1, *arg2;  
if (ARG_COUNT(ht) != 2 || getParameters(ht,2,&arg1,&arg2)==FAILURE) {  
    WRONG_PARAM_COUNT;  
}
```

NOTE: Les arguments peuvent être passés par valeur ou par référence. Dans les deux cas, vous devez passer `&(pval *)` à `getParameters`. Si vous voulez vérifier que le *n*-ième paramètre a été passé par référence ou par valeur, vous devez utiliser la fonction `ParameterPassedByReference(ht,n)`. Elle retournera 1 ou 0.

Lorsque vous modifiez l'un des paramètres, qu'ils soient envoyés par référence ou par valeur, vous pouvez le passer à `pval_destructor` pour le réinitialiser, ou, s'il s'agit d'un tableau et que vous voulez ajouter des valeurs, vous pouvez utiliser des fonctions similaires à celles qui sont dans `internal_functions.h`, qui manipule `return_value` comme tableau.

Par ailleurs, si vous modifiez un paramètre en `IS_STRING`, assurez-vous que vous avez bien assigné une nouvelle chaîne avec `estrdup()` et une nouvelle longueur de chaîne. Seulement après, vous pouvez modifier le type en `IS_STRING`. Si vous modifiez une chaîne en `IS_STRING` ou `IS_ARRAY` vous devez d'abord appeler le destructeur `pval_destructor`.

C.1.3 Fonctions à nombre d'arguments variable

[\[Notes en ligne\]](#)

Une fonction peut prendre un nombre variable d'arguments. Si votre fonction peut prendre deux ou trois arguments, utiliser la syntaxe suivante :

Fonctions à nombre d'arguments variable

```

pval *arg1, *arg2, *arg3;
int arg_count = ARG_COUNT(ht);
if (arg_count < 2 || arg_count > 3 ||
    getParameters(ht, arg_count, &arg1, &arg2, &arg3) == FAILURE) {
    WRONG_PARAM_COUNT;
}

```

C.1.4 Utiliser les arguments d'une fonction

[\[Notes en ligne\]](#)

De type de chaque argument est stocké dans le champs pval. Ce champs peut prendre les valeurs suivantes :

IS_STRING	Chaîne de caractères
IS_DOUBLE	Nombre à virgule flottante, en précision double
IS_LONG	Entier long
IS_ARRAY	Tableau
IS_EMPTY	Aucune
IS_USER_FUNCTION	??
IS_INTERNAL_FUNCTION	?? (Si ce type ne peut pas être passé à une fonction, effacez-le) @tab
IS_CLASS	??
IS_OBJECT	??

Si vous recevez un argument d'un type, te que vous voulez l'utiliser avec un autre type, ou si vous voulez simplement forcer le type, vous pouvez utiliser l'une des fonctions de conversion suivantes :

```

convert_to_long(arg1);
convert_to_double(arg1);
convert_to_string(arg1);
convert_to_boolean_long(arg1); /* Si la chaîne est "" ou "0" elle devient 0, 1 sinon */
convert_string_to_number(arg1); /* Convertit une chaîne en LONG ou DOUBLE suivant la chaîne */

```

Ces fonctions convertissent sur place : elles ne retourne aucune valeur.

La valeur de l'argument est enregistrées dans une union. Les membres sont :

- IS_STRING: arg1->value.str.val
- IS_LONG: arg1->value.lval
- IS_DOUBLE: arg1->value.dval

C.1.5 Gestion de la mémoire dans une fonction

[\[Notes en ligne\]](#)

Toute la mémoire nécessaire à une fonction doit être allouée avec emalloc() ou estrdup(). Ces fonctions ont le

gout et l'odeur des classiques malloc() et strdup(). La mémoire doit être libérée avec free().

Il y a deux types de mémoire dans ce programme : la mémoire qui est retournée à l'analyseur, et la mémoire qui est nécessaire pour le stockage temporaire dans la fonction. Lorsque vous assignez une chaîne dans une variable qui est retournée à l'analyseur, assurez-vous de bien allouer la mémoire avec emalloc() ou estrdup(). Cette mémoire ne doit JAMAIS être libérée, sauf si vous réécrivez votre original plus loin, dans la même fonction (mais ce n'est pas de la programmation propre).

Pour tous vos besoins en mémoire temporaire/permanente dont vous avez besoin dans vos fonctions/librairies, vous devez utiliser les fonctions emalloc(), estrdup() et free(). Elles se comportent EXACTEMENT comme leurs homologues. Tout ce qui est créé avec emalloc() ou estrdup() doit être libéré avec free() à un moment ou un autre, à moins que ce ne soit utile ailleurs dans le programme; sinon, il va y avoir une fuite de mémoire. La signification de "Elles se comportent EXACTEMENT comme leurs homologues" est que si vous libérez une variable qui n'a pas été créée avec emalloc() ou estrdup(), vous courez droit à un crash ("segmentation fault"). Soyez alors extrêmement prudent, et libérez toute votre mémoire inutilisée.

Si vous compilez avec "-DDEBUG", PHP 3 affichera la liste de tous les appels à emalloc() et estrdup() mais jamais à free() lorsque celui-ci intervient dans un script spécifié.

C.1.6 Affecter une variable dans la table des symboles

[\[Notes en ligne\]](#)

Un grand nombre de macros sont disponibles pour rendre plus facile l'insertion de variables dans la table des symboles :

- SET_VAR_STRING(name,value)
- SET_VAR_DOUBLE(name,value)
- SET_VAR_LONG(name,value)

Soyez prudent. La valeur doit être placée dans une portion de mémoire créée avec malloc(), sinon le gestionnaire de mémoire essaiera de libérer le pointeur plus tard. Ne passez aucune mémoire allouée statiquement à SET_VAR_STRING.

Les tables des symboles de PHP 3 est une table de hash. A n'importe quel moment, &symbol_table est un pointeur sur la table principale, et active_symbol_table pointe sur la table actuellement utilisée. (ces deux tables peuvent être identiques au démarrage, ou différentes, suivant que vous êtes dans une fonction ou non). Les exemples suivants utilisent 'active_symbol_table'. Vous devriez la remplacer par &symbol_table si vous voulez travailler sur la table principale. De plus, les mêmes fonctions peuvent être appliquées à des tableaux, comme expliqué ci-dessous.

Vérification de l'existence de \$foo dans la table des symboles

```
if (hash_exists(active_symbol_table, "foo", sizeof("foo"))) {
    // existe...
} else {
    // n'existe pas
}
```

Rechercher la taille d'une variable dans la table des symboles

```
hash_find(active_symbol_table, "foo", sizeof("foo"), &pvalue);
```

```
check(pvalue.type);
```

En PHP 3.0, les tableaux sont implémentés en utilisant les mêmes tables de hash que les variables. Cela signifie que les deux fonctions ci dessus peuvent être appelées pour vérifier la présence de variables dans un tableau.

Si vous voulez définir un nouveau tableau dans la table des symboles, utilisez le code suivant.

D'abord, vous devez vérifier qu'il n'existe pas, avec `hash_exists()` ou `hash_find()`.

Puis, initialisez le tableau :

Initialisation d'un tableau

```
pval arr;
if (array_init(&arr) == FAILURE) { failed... };
hash_update(active_symbol_table, "foo", sizeof("foo"), &arr, sizeof(pval), NULL);
```

Ce code déclare un nouveau tableau, appelé \$foo, dans la table de symbole. Ce tableau est vide.

Voici comment ajouter deux nouvelles entrées dans ce tableau :

Ajout d'entrées dans un tableau.

```
pval entry;
entry.type = IS_LONG;
entry.value.lval = 5;
/* définit $foo["bar"] = 5 */
hash_update(arr.value.ht, "bar", sizeof("bar"), &entry, sizeof(pval), NULL);
/* définit $foo[7] = 5 */
hash_index_update(arr.value.ht, 7, &entry, sizeof(pval), NULL);
/* définit la prochaine place libre dans $foo[],
 * $foo[8], qui sera 5 (comme en php2)
 */
hash_next_index_insert(arr.value.ht, &entry, sizeof(pval), NULL);
```

Si vous voulez modifier une valeur que vous avez inséré dans une table de hash, vous devez d'abord la lire dans la table. Pour éviter cette recherche, vous pouvez fournir une pval ** à la fonction d'ajout dans la table de hash, et elle modifiera la valeur à l'adresse pval *, avec la valeur donnée. Si cette valeur est NULL, (comme dans tous les exemples ci dessus), ce paramètre sera ignoré.

`hash_next_index_insert()` utiliser plus ou moins la même logique que "`$foo[] = bar;`" in PHP 2.0.

Si vous construisez un tableau, pour le retourner, vous pouvez l'initialiser comme ceci :

```
if (array_init(return_value) == FAILURE) { échec...; }
```

...puis ajouter les valeurs grâce aux macros:

```
add_next_index_long(return_value, long_value);
add_next_index_double(return_value, double_value);
add_next_index_string(return_value, estrdup(string_value));
```

Bien sur, si l'ajout n'est pas fait juste après l'initialisation, vous devrez d'abord rechercher le tableau :

```
pval *arr;
if (hash_find(active_symbol_table, "foo", sizeof("foo"), (void **) &arr) == FAILURE)
{ introuvable... }
else
{ utilisez arr->value.ht... }
```

Notez que `hash_find` reçoit un pointeur sur un pointeur sur `pval`, et pas un pointeur sur `pval`. Toutes les fonctions d'accès aux hash retournent `TRUE` (SUCCES) ou `FALSE` (FAILURE), excepté `hash_exists()`, qui retourne un booléen.

C.1.7 Retourne une valeur simple

[\[Notes en ligne\]](#)

Un grand nombre de macros sont disponible pour simplifier le retour des valeurs. La macro `RETURN_*` fixe la valeur de retour, et termine la fonction :

- `RETURN`
- `RETURN_FALSE`
- `RETURN_TRUE`
- `RETURN_LONG(l)`
- `RETURN_STRING(s,dup)` Si `dup` est `TRUE`, duplique la chaîne.
- `RETURN_STRINGL(s,l,dup)` retourne la chaîne (`s`) en spécifiant la longueur (`l`).
- `RETURN_DOUBLE(d)`

La macro `RETVAL_*` macros fixe la valeur de retour, mais ne termine pas la fonction.

- `RETVAL_FALSE`
- `RETVAL_TRUE`
- `RETVAL_LONG(l)`
- `RETVAL_STRING(s,dup)` Si `dup` est `TRUE`, duplique la chaîne
- `RETVAL_STRINGL(s,l,dup)` retourne la chaîne (`s`) en spécifiant la longueur (`l`).
- `RETVAL_DOUBLE(d)`

Les macros ci dessus vont utiliser `estrdup()` sur les arguments passés. Cela vous permet de libérer tranquillement les arguments après avoir appelé cette fonction, ou bien, utiliser de la mémoire allouée statiquement.

Si votre fonction retourne un booléen de succès/erreur, utilisez toujours `RETURN_TRUE` et `RETURN_FALSE` respectivement.

C.1.8 Retourner des valeurs complexes

[\[Notes en ligne\]](#)

Votre fonction peut aussi retourner des valeurs complexes, tels que des objets ou tableaux. Retourner un objet:

1. Appeler `object_init(return_value)`.
2. Remplissez les valeurs. Les fonctions utilisables sont listées ci dessous.
3. Eventuellement, enregistrez les fonctions pour cet objet. Afin de lire des valeurs de cet objet, la fonction doit lire dans "this", dans la table de symbole active `active_symbol_table`. Son type doit être `IS_OBJECT`, et c'est une table de hash basique. (i.e., vous pouvez utiliser les fonctions habituelles de `.value.ht`). L'enregistrement réel peut être fait comme suit :

```
add_method( return_value, function_name, function_ptr );
```

Les fonctions d'accès aux objets sont :

- `add_property_long(return_value, property_name, l)` – Ajoute un membre nommé 'property_name', de type long, égal à 'l'
- `add_property_double(return_value, property_name, d)` – Idem, ajoute un double
- `add_property_string(return_value, property_name, str)` – Idem, ajoute une chaîne
- `add_property_stringl(return_value, property_name, str, l)` – Idem, ajoute une chaîne de longueur 'l'.

Retournez un tableau :

1. Appelez `array_init(return_value)`.
2. Remplissez les valeurs. Les fonctions disponibles sont listées ci dessous.

Les fonctions utilisées pour accéder à un tableau sont :

- `add_assoc_long(return_value,key,l)` – Ajoute une entrée associative avec la clé 'key' et la valeur 'l', de type long
- `add_assoc_double(return_value,key,d)` – Ajoute une entrée associative avec la clé 'key' et la valeur 'l', de type double
- `add_assoc_string(return_value,key,str,duplicate)`
- `add_assoc_stringl(return_value,key,str,length,duplicate)` specify the string length
- `add_index_long(return_value,index,l)` – Ajoute une entrée d'index 'index' avec la valeur 'l', de type long
- `add_index_double(return_value,index,d)`
- `add_index_string(return_value,index,str)`
- `add_index_stringl(return_value,index,str,length)` – spécifie la longueur de la chaîne.
- `add_next_index_long(return_value,l)` – ajoute une entrée tableau, dans le prochain offset libre, de longueur 'l', de type long
- `add_next_index_double(return_value,d)`
- `add_next_index_string(return_value,str)`
- `add_next_index_stringl(return_value,str,length)` – specify the string length

C.1.9 Using the resource list

[\[Notes en ligne\]](#)

PHP 3.0 dispose de standard pour traiter un certains nombre de ressources. Ils remplacent tous les listes de PHP 2.0.

Fonctions accessibles :

- `php3_list_insert(ptr, type)` – retourne l'identifiant 'id' de la nouvelle ressource insérée.
- `php3_list_delete(id)` – efface la ressource d'identifiant id
- `php3_list_find(id,*type)` – retourne le pointeur de la ressource d'identifiant id, et modifie le type 'type'

Typiquement, ces fonctions sont utilisées pour les pilotes SQL, mais elles peuvent servir n'importe quoi d'autre. Par exemple, conserver un pointeur de fichier.

La liste standard de code ressemble à ceci :

Ajouter une nouvelle ressource

```
RESOURCE *resource;
/* ...alloue de la mémoire pour la ressource, et l'acquiert ... */
/* Ajoute la nouvelle ressource dans la liste */
return_value->value.lval = php3_list_insert((void *) resource, LE_RESOURCE_TYPE);
return_value->type = IS_LONG;
```

Utiliser une ressource existante

```
pval *resource_id;
RESOURCE *resource;
int type;
convert_to_long(resource_id);
resource = php3_list_find(resource_id->value.lval, &type);
if (type != LE_RESOURCE_TYPE) {
    php3_error(E_WARNING, "resource index %d has the wrong type", resource_id->value.lval);
    RETURN_FALSE;
}
/* ...utiliser la ressource... */
```

Effacer une ressource existante

```
pval *resource_id;
RESOURCE *resource;
int type;
convert_to_long(resource_id);
php3_list_delete(resource_id->value.lval);
```

Les types de ressources doivent être enregistré dans le fichier `php3_list.h`, dans l'énumération `list_entry_type`. En plus, il faut penser à ajouter une fonction de terminaison, pour chaque type de ressource défini, dans le fichier `list.c`, pour la fonction `list_entry_destructor()` (même si vous n'avez rien de particulier à faire lors de la terminaison, vous devez au moins ajouter un cas vide).

C.1.10 Utiliser la table des ressources persistantes.

[\[Notes en ligne\]](#)

PHP 3.0 dispose d'un lieu de stockage des ressources persistantes (i.e., les ressources qui doivent être conservées d'un hit à l'autre). Le premier module à utiliser cette capacité a été MySQL, et mSQL suivi, ce qui fait que l'on peut se faire une impression du fonctionnement de cette fonction avec `mysql.c`. Les fonctions ressemblent à ceci :

- `php3_mysql_do_connect`
- `php3_mysql_connect()`
- `php3_mysql_pconnect()`

L'idée conductrice de ces modules est la suivante :

1. Programmez tout votre module pour qu'il travaille avec les ressources standard, comme mentionné dans la section (9).
2. Ajoutez une autre fonction de connexion, qui vérifie d'abord que la ressource existe dans la liste des ressources persistantes. Si c'est le cas, enregistrez cette ressource comme pour les ressources standard (et grâce à la première étape, cela va fonctionner immédiatement). Si la ressource n'existe pas, créez-la, ajoutez-la à la liste de ressources persistantes, et ajoutez-la à la liste de ressources, ce qui fait que le code va fonctionner, et que le prochain appel renverra une ressource existante. Vous devez enregistrer ces fonctions avec un type différent (LE_MYSQL_LINK pour les liens non persistants, et LE_MYSQL_PLINK pour les liens persistants).

Si vous jetez un oeil dans `mysql.c`, vous verrez que, hormis la fonction de connexion complexe, rien n'a dû être changé dans le module.

La même interface existe pour la liste des ressources standard, et pour la liste des ressources persistantes, seule la 'list' est remplacée par 'plist':

- `php3_plist_insert(ptr, type)` – retourne l'identifiant 'id' de la nouvelle ressource insérée.
- `php3_plist_delete(id)` – efface la ressource d'identifiant id
- `php3_plist_find(id, *type)` – retourne le pointeur de la ressource d'identifiant id, et modifie le type 'type'

Cependant, il est probable que ces fonctions seront inutiles pour vous, lorsque vous essayerez d'implémenter un module persistant. Typiquement, on utilise le fait que la liste de ressources persistantes est une table de hash. Par exemple, dans les modules MySQL/mSQL, lors d'un appel à `pconnect()`, la fonction construit une chaîne avec l'hôte/utilisateur/mot_de_passe, et l'utilise pour enregistrer dans la table de hash. Au prochain appel, avec les mêmes hôte/utilisateur/mot_de_passe, la même clé sera générée, et la ressource associée sera retrouvée.

Jusqu'à ce que la documentation s'étoffe, jetez un oeil aux fichiers `mysql.c` ou `mssql.c` pour voir comment implémenter vos accès aux ressources persistantes.

Une chose importante à noter : les ressources qui sont enregistrées dans la liste de ressource persistante ne DOIVENT PAS être allouées avec le gestionnaire de mémoire PHP, c'est à dire qu'elles ne doivent pas être créées avec `emalloc()`, `estrdup()`, etc. Au contraire, il faut utiliser les fonctions standard `malloc()`, `strdup()`, etc. La raison est fort simple : à la fin de la requête, la mémoire sera supprimée par le gestionnaire. Étant donné que les liens persistants doivent être conservés, il ne faut pas utiliser le gestionnaire de mémoire.

Lorsque vous enregistrez une ressource qui sera placée dans la liste de ressources persistantes, il faut ajouter les destructeurs dans les deux listes de ressources, persistantes ou pas. Le destructeur de la liste de ressources non persistantes ne doit rien faire du tout, tandis que celui de la liste de ressources persistantes doit libérer proprement toutes les ressources acquises (mémoire, lien SQL...). Comme pour les ressources non persistantes vous DEVEZ ajouter un destructeur, même si il ne fait rien. N'oubliez pas que `emalloc()` et compagnie ne doivent pas être utilisés en conjonction avec la liste de ressources persistantes, et donc, vous ne devez pas utiliser `efree()` non plus.

C.1.11 Ajouter des directives de configuration à l'exécution

[\[Notes en ligne\]](#)

De nombreuses caractéristiques de PHP 3 peuvent être configurées à l'exécution. Ces directives peuvent apparaître dans le fichier ``php3.ini'`, ou, dans le cas du module Apache, dans le fichier ``conf'`. L'avantage de l'avoir dans le fichier ``conf'`, est que ces caractéristiques peuvent être configurées dossier par dossier. Cela signifie qu'un dossier peut avoir un `safe mode exec dir`, tandis qu'un autre en aura un autre. Cette granularité de la configuration peut être extrêmement pratique lorsque le serveur supporte plusieurs

serveurs virtuels.

Les étapes de configuration d'une nouvelle directive sont :

1. Ajouter la directive à la structure `php3_ini_structure` dans le fichier ``mod_php3.h'`.
2. Dans `main.c`, éditez la fonction `php3_module_startup` et ajoutez l'appel approprié à `cfg_get_string()` ou `cfg_get_long()`.
3. Ajoutez la directive, ses restrictions et un commentaire dans la structure `php3_commands` du fichier `mod_php3.c`. Notez la partie restrictions `RSRC_CONF` sont des directives qui ne peuvent être disponibles que dans le fichier de configuration Apache. Toutes les directives `OR_OPTIONS` peuvent être placées n'importe où, y compris dans un fichier ``.htaccess'`.
4. Soit dans `php3take1handler()`, soit dans `php3flaghandler()`, ajoutez l'entrée appropriée pour votre directive.
5. Dans la section de configuration, de `_php3_info()`, dans le fichier `functions/info.c`, vous devez ajouter votre configuration.
6. Finalement, vous devez utiliser votre configuration quelque part. Elle sera accessible par `php3_ini.directive`.

C.2 Appeler des fonctions utilisateurs

[\[Notes en ligne\]](#)

Pour appeler des fonctions utilisateurs depuis une fonction interne, vous devez utiliser la fonction `call_user_function()`.

`call_user_function()` retourne `SUCCESS` en cas de succès, et `FAILURE` en cas d'échec, ou si la fonction n'a pas été trouvée. Vous devez vérifier cette valeur. Si la réponse est `SUCCESS`, vous êtes responsable de la destruction de `retval` (ou alors, retournez la comme valeur de réponse de votre fonction). Si la réponse est `FAILURE`, la valeur de `retval` est indéfinie, et vous ne devez pas y toucher.

Toutes les fonctions internes qui appellent une fonction utilisateur, *DOIVENT* être réentrante. En particulier, elles ne doivent pas utiliser de valeurs globales, ou de variables statiques.

`call_user_function()` prend 6 arguments :

C.2.1 HashTable *function_table

[\[Notes en ligne\]](#)

La table de hash dans laquelle la fonction doit être recherchée.

C.2.2 pval *object

[\[Notes en ligne\]](#)

Un pointeur sur un objet sur lequel la fonction est invoquée. Il devrait être à `NULL`, si on invoque une fonction globale. Si il n'est pas à `NULL` (ie, il pointe sur un objet), l'argument `function_table` est ignorée, et la liste des fonctions sera lue dans l'objet, plutôt que dans l'argument. L'objet PEUT être modifié par la fonction qui est appelée (la fonction y aura accès via `$this`). Si, pour quelque raison, vous ne le voulez pas, envoyez une copie de l'objet à la place.

C.2.3 pval *function_name

[\[Notes en ligne\]](#)

Le nom de la fonction à appeler. Elle doit être de type pval, IS_STRING, avec les valeurs de function_name.str.val et function_name.str.len correctes. function_name est modifié par call_user_function() – il est converti en minuscule. Si vous voulez préserver la casse, envoyez une copie du nom de la fonction.

C.2.4 pval *retval

[\[Notes en ligne\]](#)

Un pointeur sur une structure pval, dans laquelle la valeur de retour de la fonction sera placée. La structure doit avoir été allouée au préalable, – call_user_function() ne l'allouera pas.

C.2.5 int param_count

[\[Notes en ligne\]](#)

Le nombre de paramètre passé à la fonction.

C.2.6 pval *params[]

[\[Notes en ligne\]](#)

Un tableau de pointeur sur les valeurs qui vont être passées comme arguments à la fonction. Le premier argument est à l'offset 0, le second à l'offset 1,... Le tableau est un tableau de pointeurs sur pval; Les pointeurs sont envoyés tels quels à la fonction, ce qui signifie que si la fonction modifie les arguments, les valeurs originales seront modifiées. Si vous voulez l'éviter, passez une copie à la place.

C.3 Rapport d'erreurs

[\[Notes en ligne\]](#)

Pour signaler les erreurs d'une fonction interne, vous devez appeler la fonction php3_error(). Cette fonction prend deux arguments au moins : le niveau de l'erreur, et le message d'erreur, sous forme de chaîne de caractères. Tous les arguments suivants sont des paramètres de formats de chaîne. Les niveaux d'erreurs sont :

C.3.1 E_NOTICE

[\[Notes en ligne\]](#)

Les notes ne sont pas affichées par défaut, et indique que le script a rencontré quelque chose qui peut être une erreur, mais peut aussi être un événement normal dans la vie du script. Par exemple, essayer d'accéder à une valeur qui n'a pas été déclarée, ou appeler [stat\(\)](#) sur un fichier qui n'existe pas.

C.3.2 E_WARNING

[\[Notes en ligne\]](#)

Les alertes sont affichées par défaut, mais n'interrompent pas l'exécution du script. Elles indiquent un problème qui doit être intercepté par le script avant que l'appel. Par exemple, appeler [ereg\(\)](#) avec une regex invalide.

C.3.3 E_ERROR

[\[Notes en ligne\]](#)

Les erreurs sont aussi affichées par défaut, et l'exécution du script est interrompue. Elles indiquent des erreurs qui ne peuvent pas être ignorées, comme des problèmes d'allocation de mémoire, par exemple.

C.3.4 E_PARSE

[\[Notes en ligne\]](#)

Les erreurs d'analyse doivent être générées que par l'analyseur. Elles ne sont citées ici que dans le but d'être exhaustif.

C.3.5 E_CORE_ERROR

[\[Notes en ligne\]](#)

Elles sont similaires aux erreurs E_ERROR, mais elles sont générées par le code de PHP. Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.6 E_CORE_WARNING

[\[Notes en ligne\]](#)

Elles sont similaires à E_WARNING, mais elles sont générées par le code de PHP. Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.7 E_COMPILE_ERROR

[\[Notes en ligne\]](#)

Elles sont similaires à E_ERROR, mais elles sont générées par Zend Scripting Engine. Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.8 E_COMPILE_WARNING

[\[Notes en ligne\]](#)

Elles sont similaires à E_WARNING, mais elles sont générées par Zend Scripting Engine. Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.9 E_USER_ERROR

[\[Notes en ligne\]](#)

E_USER_ERROR est comparable à E_ERROR. Elle est générée en PHP par l'utilisation de la fonction [trigger_error\(\)](#). Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.10 E_USER_WARNING

[\[Notes en ligne\]](#)

E_USER_WARNING est comparable à E_WARNING. Elle est générée en PHP par l'utilisation de la fonction [trigger_error\(\)](#). Les fonctions ne doivent pas générer ce genre d'erreur.

C.3.11 E_USER_NOTICE

[\[Notes en ligne\]](#)

E_USER_WARNING est comparable à E_NOTICE. Elle est générée en PHP par l'utilisation de la fonction [trigger_error\(\)](#). Les fonctions ne doivent pas générer ce genre d'erreur.

Index des fonctions

[\[Notes en ligne\]](#) [\[Exemples\]](#)

A

7. [abs](#)
8. [accept_connect](#)
9. [acos](#)
10. [addslashes](#)
11. [addslashes](#)
12. [apache_lookup_uri](#)
13. [apache_note](#)
14. [array](#)
15. [array_count_values](#)
16. [array_diff](#)
17. [array_flip](#)
18. [array_intersect](#)
19. [array_keys](#)
20. [array_merge](#)
21. [array_merge_recursive](#)
22. [array_multisort](#)
23. [array_pad](#)
24. [array_pop](#)
25. [array_push](#)
26. [array_rand](#)
27. [array_reverse](#)
28. [array_shift](#)
29. [array_slice](#)
30. [array_splice](#)
31. [array_unique](#)
32. [array_unshift](#)
33. [array_values](#)
34. [array_walk](#)
35. [arsort](#)
36. [ascii2ebcdic](#)
37. [asin](#)
38. [asort](#)
39. [aspell_check](#)
40. [aspell_check--raw](#)
41. [aspell_new](#)
42. [aspell_suggest](#)
43. [assert](#)
44. [assert--options](#)
45. [atan](#)
46. [atan2](#)

B

- 47. [base64_decode](#)
- 48. [base64_encode](#)
- 49. [base_convert](#)
- 50. [basename](#)
- 51. [bcadd](#)
- 52. [bccomp](#)
- 53. [bcdiv](#)
- 54. [bcmod](#)
- 55. [bcmul](#)
- 56. [bcpow](#)
- 57. [bcscale](#)
- 58. [bcsqrt](#)
- 59. [bcsub](#)
- 60. [bin2hex](#)
- 61. [bind](#)
- 62. [bindec](#)
- 63. [bindtextdomain](#)

C

- 64. [call_user_func](#)
- 65. [call_user_method](#)
- 66. [ceil](#)
- 67. [chdir](#)
- 68. [checkdate](#)
- 69. [checkdnsrr](#)
- 70. [chgrp](#)
- 71. [chmod](#)
- 72. [chop](#)
- 73. [chown](#)
- 74. [chr](#)
- 75. [chunk_split](#)
- 76. [class_exists](#)
- 77. [clearstatcache](#)
- 78. [close](#)
- 79. [closedir](#)
- 80. [closelog](#)
- 81. [com_get](#)
- 82. [com_invoke](#)
- 83. [com_load](#)
- 84. [com_propget](#)
- 85. [com_propput](#)
- 86. [com_propset](#)
- 87. [com_set](#)
- 88. [compact](#)
- 89. [connect](#)
- 90. [connection_aborted](#)
- 91. [connection_status](#)

92. [connection timeout](#)
93. [convert_cyr_string](#)
94. [copy](#)
95. [cos](#)
96. [count](#)
97. [count_chars](#)
98. [cpdf_add_annotation](#)
99. [cpdf_add_outline](#)
100. [cpdf_arc](#)
101. [cpdf_begin_text](#)
102. [cpdf_circle](#)
103. [cpdf_clip](#)
104. [cpdf_close](#)
105. [cpdf_closepath](#)
106. [cpdf_closepath_fill_stroke](#)
107. [cpdf_closepath_stroke](#)
108. [cpdf_continue_text](#)
109. [cpdf_curveto](#)
110. [cpdf_end_text](#)
111. [cpdf_fill](#)
112. [cpdf_fill_stroke](#)
113. [cpdf_finalize](#)
114. [cpdf_finalize_page](#)
115. [cpdf_global_set_document_limits](#)
116. [cpdf_import_jpeg](#)
117. [cpdf_lineto](#)
118. [cpdf_moveto](#)
119. [cpdf_newpath](#)
120. [cpdf_open](#)
121. [cpdf_output_buffer](#)
122. [cpdf_page_init](#)
123. [cpdf_place_inline_image](#)
124. [cpdf_rect](#)
125. [cpdf_restore](#)
126. [cpdf_rlineto](#)
127. [cpdf_rmoveto](#)
128. [cpdf_rotate](#)
129. [cpdf_save](#)
130. [cpdf_save_to_file](#)
131. [cpdf_scale](#)
132. [cpdf_set_char_spacing](#)
133. [cpdf_set_creator](#)
134. [cpdf_set_current_page](#)
135. [cpdf_set_font](#)
136. [cpdf_set_horiz_scaling](#)
137. [cpdf_set_keywords](#)
138. [cpdf_set_leading](#)
139. [cpdf_set_page_animation](#)
140. [cpdf_set_subject](#)
141. [cpdf_set_text_matrix](#)
142. [cpdf_set_text_pos](#)

- 143. [cpdf_set_text_rendering](#)
- 144. [cpdf_set_text_rise](#)
- 145. [cpdf_set_title](#)
- 146. [cpdf_set_word_spacing](#)
- 147. [cpdf_setdash](#)
- 148. [cpdf_setflat](#)
- 149. [cpdf_setgray](#)
- 150. [cpdf_setgray_fill](#)
- 151. [cpdf_setgray_stroke](#)
- 152. [cpdf_setlinecap](#)
- 153. [cpdf_setlinejoin](#)
- 154. [cpdf_setlinewidth](#)
- 155. [cpdf_setmiterlimit](#)
- 156. [cpdf_setrgbcolor](#)
- 157. [cpdf_setrgbcolor_fill](#)
- 158. [cpdf_setrgbcolor_stroke](#)
- 159. [cpdf_show](#)
- 160. [cpdf_show_xy](#)
- 161. [cpdf_stringwidth](#)
- 162. [cpdf_stroke](#)
- 163. [cpdf_text](#)
- 164. [cpdf_translate](#)
- 165. [crc32](#)
- 166. [create_function](#)
- 167. [crypt](#)
- 168. [curl_close](#)
- 169. [curl_exec](#)
- 170. [curl_init](#)
- 171. [curl_setopt](#)
- 172. [curl_version](#)
- 173. [current](#)
- 174. [cybercash_base64_decode](#)
- 175. [cybercash_base64_encode](#)
- 176. [cybercash_decr](#)
- 177. [cybercash_encr](#)

D

- 178. [date](#)
- 179. [dba_close](#)
- 180. [dba_delete](#)
- 181. [dba_exists](#)
- 182. [dba_fetch](#)
- 183. [dba_firstkey](#)
- 184. [dba_insert](#)
- 185. [dba_nextkey](#)
- 186. [dba_open](#)
- 187. [dba_optimize](#)
- 188. [dba_popen](#)
- 189. [dba_replace](#)

- 190. [dba_sync](#)
- 191. [dbase_add_record](#)
- 192. [dbase_close](#)
- 193. [dbase_create](#)
- 194. [dbase_delete_record](#)
- 195. [dbase_get_record](#)
- 196. [dbase_get_record_with_names](#)
- 197. [dbase_numfields](#)
- 198. [dbase_numrecords](#)
- 199. [dbase_open](#)
- 200. [dbase_pack](#)
- 201. [dbase_replace_record](#)
- 202. [dblist](#)
- 203. [dbmclose](#)
- 204. [dbmdelete](#)
- 205. [dbmexists](#)
- 206. [dbmfetch](#)
- 207. [dbmfirstkey](#)
- 208. [dbminsert](#)
- 209. [dbmnextkey](#)
- 210. [dbmopen](#)
- 211. [dbmreplace](#)
- 212. [dcgettext](#)
- 213. [debugger_off](#)
- 214. [debugger_on](#)
- 215. [decbin](#)
- 216. [dechex](#)
- 217. [decoct](#)
- 218. [define](#)
- 219. [define_syslog_variables](#)
- 220. [defined](#)
- 221. [deg2rad](#)
- 222. [delete](#)
- 223. [dgettext](#)
- 224. [die](#)
- 225. [dir](#)
- 226. [dirname](#)
- 227. [diskfreespace](#)
- 228. [dl](#)
- 229. [doubleval](#)

E

- 230. [each](#)
- 231. [easter_date](#)
- 232. [easter_days](#)
- 233. [ebcdic2ascii](#)
- 234. [echo](#)
- 235. [empty](#)
- 236. [end](#)

- 237. [ereg](#)
- 238. [ereg_replace](#)
- 239. [eregi](#)
- 240. [eregi_replace](#)
- 241. [error_log](#)
- 242. [error_reporting](#)
- 243. [escapeshellarg](#)
- 244. [escapeshellcmd](#)
- 245. [eval](#)
- 246. [exec](#)
- 247. [exit](#)
- 248. [exp](#)
- 249. [explode](#)
- 250. [extension_loaded](#)
- 251. [extract](#)
- 252. [ezmlm_hash](#)

F

- 253. [fclose](#)
- 254. [fdf_close](#)
- 255. [fdf_create](#)
- 256. [fdf_get_file](#)
- 257. [fdf_get_status](#)
- 258. [fdf_get_value](#)
- 259. [fdf_next_field_name](#)
- 260. [fdf_open](#)
- 261. [fdf_save](#)
- 262. [fdf_set_ap](#)
- 263. [fdf_set_file](#)
- 264. [fdf_set_flags](#)
- 265. [fdf_set_javascript_action](#)
- 266. [fdf_set_opt](#)
- 267. [fdf_set_status](#)
- 268. [fdf_set_submit_form_action](#)
- 269. [fdf_set_value](#)
- 270. [feof](#)
- 271. [fflush](#)
- 272. [fgetc](#)
- 273. [fgetcsv](#)
- 274. [fgets](#)
- 275. [fgetss](#)
- 276. [file](#)
- 277. [file_exists](#)
- 278. [fileatime](#)
- 279. [filectime](#)
- 280. [filegroup](#)
- 281. [fileinode](#)
- 282. [filemtime](#)
- 283. [fileowner](#)

- 284. [fileperms](#)
- 285. [filepro](#)
- 286. [filepro_fieldcount](#)
- 287. [filepro_fieldname](#)
- 288. [filepro_fieldtype](#)
- 289. [filepro_fieldwidth](#)
- 290. [filepro_retrieve](#)
- 291. [filepro_rowcount](#)
- 292. [filesize](#)
- 293. [filetype](#)
- 294. [flock](#)
- 295. [floor](#)
- 296. [flush](#)
- 297. [fopen](#)
- 298. [fpassthru](#)
- 299. [fputs](#)
- 300. [fread](#)
- 301. [frenchtojd](#)
- 302. [fscanf](#)
- 303. [fseek](#)
- 304. [fsockopen](#)
- 305. [fstat](#)
- 306. [ftell](#)
- 307. [ftp_cdup](#)
- 308. [ftp_chdir](#)
- 309. [ftp_connect](#)
- 310. [ftp_delete](#)
- 311. [ftp_fget](#)
- 312. [ftp_fput](#)
- 313. [ftp_get](#)
- 314. [ftp_login](#)
- 315. [ftp_mdtm](#)
- 316. [ftp_mkdir](#)
- 317. [ftp_nlist](#)
- 318. [ftp_pasv](#)
- 319. [ftp_put](#)
- 320. [ftp_pwd](#)
- 321. [ftp_quit](#)
- 322. [ftp_rawlist](#)
- 323. [ftp_rename](#)
- 324. [ftp_rmdir](#)
- 325. [ftp_site](#)
- 326. [ftp_size](#)
- 327. [ftp_systype](#)
- 328. [ftruncate](#)
- 329. [func_get_arg](#)
- 330. [func_get_args](#)
- 331. [func_num_args](#)
- 332. [function_exists](#)
- 333. [fwrite](#)

G

- 334. [get_browser](#)
- 335. [get_cfg_var](#)
- 336. [get_class](#)
- 337. [get_class_methods](#)
- 338. [get_class_vars](#)
- 339. [get_current_user](#)
- 340. [get_declared_classes](#)
- 341. [get_extension_funcs](#)
- 342. [get_html_translation_table](#)
- 343. [get_included_files](#)
- 344. [get_loaded_extensions](#)
- 345. [get_magic_quotes_gpc](#)
- 346. [get_magic_quotes_runtime](#)
- 347. [get_meta_tags](#)
- 348. [get_object_vars](#)
- 349. [get_parent_class](#)
- 350. [get_required_files](#)
- 351. [getallheaders](#)
- 352. [getcwd](#)
- 353. [getdate](#)
- 354. [getenv](#)
- 355. [gethostbyaddr](#)
- 356. [gethostbyname](#)
- 357. [gethostbyname_l](#)
- 358. [getimagesize](#)
- 359. [getlastmod](#)
- 360. [getmxrr](#)
- 361. [getmyinode](#)
- 362. [getmypid](#)
- 363. [getmyuid](#)
- 364. [getprotobyname](#)
- 365. [getprotobyname](#)
- 366. [getrandmax](#)
- 367. [getrusage](#)
- 368. [getservbyname](#)
- 369. [getservbyport](#)
- 370. [gettext](#)
- 371. [gettimeofday](#)
- 372. [gettype](#)
- 373. [gmdate](#)
- 374. [gmmktime](#)
- 375. [gmp_abs](#)
- 376. [gmp_add](#)
- 377. [gmp_and](#)
- 378. [gmp_clrbit](#)
- 379. [gmp_cmp](#)
- 380. [gmp_div](#)
- 381. [gmp_div_q](#)

- 382. [gmp_div_qr](#)
- 383. [gmp_div_r](#)
- 384. [gmp_divexact](#)
- 385. [gmp_fact](#)
- 386. [gmp_gcd](#)
- 387. [gmp_gcdext](#)
- 388. [gmp_hamdist](#)
- 389. [gmp_init](#)
- 390. [gmp_intval](#)
- 391. [gmp_invert](#)
- 392. [gmp_jacobi](#)
- 393. [gmp_legendre](#)
- 394. [gmp_mod](#)
- 395. [gmp_mul](#)
- 396. [gmp_neg](#)
- 397. [gmp_or](#)
- 398. [gmp_perfect_square](#)
- 399. [gmp_popcount](#)
- 400. [gmp_pow](#)
- 401. [gmp_powm](#)
- 402. [gmp_prob_prime](#)
- 403. [gmp_random](#)
- 404. [gmp_scan0](#)
- 405. [gmp_scan1](#)
- 406. [gmp_setbit](#)
- 407. [gmp_sign](#)
- 408. [gmp_sqrt](#)
- 409. [gmp_sqrtrm](#)
- 410. [gmp_strval](#)
- 411. [gmp_sub](#)
- 412. [gmp_xor](#)
- 413. [gmstrftime](#)
- 414. [gregoriantojd](#)
- 415. [gzclose](#)
- 416. [gzcompress](#)
- 417. [gzeof](#)
- 418. [gzfile](#)
- 419. [gzgetc](#)
- 420. [gzgets](#)
- 421. [gzgetss](#)
- 422. [gzopen](#)
- 423. [gzpassthru](#)
- 424. [gzputs](#)
- 425. [gzread](#)
- 426. [gzrewind](#)
- 427. [gzseek](#)
- 428. [gztell](#)
- 429. [gzuncompress](#)
- 430. [gzwrite](#)

H

- 431. [header](#)
- 432. [headers_sent](#)
- 433. [hebreve](#)
- 434. [hebrevc](#)
- 435. [hexdec](#)
- 436. [highlight_file](#)
- 437. [highlight_string](#)
- 438. [htmlentities](#)
- 439. [htmlspecialchars](#)
- 440. [hw_array2objrec](#)
- 441. [hw_children](#)
- 442. [hw_childrenobj](#)
- 443. [hw_close](#)
- 444. [hw_connect](#)
- 445. [hw_cp](#)
- 446. [hw_deleteobject](#)
- 447. [hw_docbyanchor](#)
- 448. [hw_docbyanchorobj](#)
- 449. [hw_documentattributes](#)
- 450. [hw_documentbodytag](#)
- 451. [hw_documentcontent](#)
- 452. [hw_documentsetcontent](#)
- 453. [hw_documentsize](#)
- 454. [hw_edittext](#)
- 455. [hw_error](#)
- 456. [hw_errormsg](#)
- 457. [hw_free_document](#)
- 458. [hw_getanchors](#)
- 459. [hw_getanchorsobj](#)
- 460. [hw_getandlock](#)
- 461. [hw_getchildcoll](#)
- 462. [hw_getchildcollobj](#)
- 463. [hw_getchilddoccoll](#)
- 464. [hw_getchilddoccollobj](#)
- 465. [hw_getobject](#)
- 466. [hw_getobjectbyquery](#)
- 467. [hw_getobjectbyquerycoll](#)
- 468. [hw_getobjectbyquerycollobj](#)
- 469. [hw_getobjectbyqueryobj](#)
- 470. [hw_getparents](#)
- 471. [hw_getparentsobj](#)
- 472. [hw_getremote](#)
- 473. [hw_getremotechildren](#)
- 474. [hw_getsrcbydestobj](#)
- 475. [hw_gettext](#)
- 476. [hw_identify](#)
- 477. [hw_incollections](#)
- 478. [hw_info](#)

- 479. [hw_inscoll](#)
- 480. [hw_insdock](#)
- 481. [hw_insertdocument](#)
- 482. [hw_insertobject](#)
- 483. [hw_mapid](#)
- 484. [hw_modifyobject](#)
- 485. [hw_mv](#)
- 486. [hw_new_document](#)
- 487. [hw_objrec2array](#)
- 488. [hw_outputdocument](#)
- 489. [hw_pconnect](#)
- 490. [hw_pipedocument](#)
- 491. [hw_root](#)
- 492. [hw_unlock](#)
- 493. [hw_username](#)
- 494. [hw_who](#)

I

- 495. [ibase_close](#)
- 496. [ibase_commit](#)
- 497. [ibase_connect](#)
- 498. [ibase_errmsg](#)
- 499. [ibase_execute](#)
- 500. [ibase_fetch_object](#)
- 501. [ibase_fetch_row](#)
- 502. [ibase_field_info](#)
- 503. [ibase_free_query](#)
- 504. [ibase_free_result](#)
- 505. [ibase_num_fields](#)
- 506. [ibase_pconnect](#)
- 507. [ibase_prepare](#)
- 508. [ibase_query](#)
- 509. [ibase_rollback](#)
- 510. [ibase_timefmt](#)
- 511. [ibase_trans](#)
- 512. [icap_close](#)
- 513. [icap_delete_event](#)
- 514. [icap_fetch_event](#)
- 515. [icap_list_alarms](#)
- 516. [icap_list_events](#)
- 517. [icap_open](#)
- 518. [icap_snooze](#)
- 519. [icap_store_event](#)
- 520. [ifx_affected_rows](#)
- 521. [ifx_blobinfile_mode](#)
- 522. [ifx_byteasvarchar](#)
- 523. [ifx_close](#)
- 524. [ifx_connect](#)
- 525. [ifx_copy_blob](#)

I

- 526. [ifx_create_blob](#)
- 527. [ifx_create_char](#)
- 528. [ifx_do](#)
- 529. [ifx_error](#)
- 530. [ifx_errormsg](#)
- 531. [ifx_fetch_row](#)
- 532. [ifx_fieldproperties](#)
- 533. [ifx_fielddtypes](#)
- 534. [ifx_free_blob](#)
- 535. [ifx_free_char](#)
- 536. [ifx_free_result](#)
- 537. [ifx_free_slob](#)
- 538. [ifx_get_blob](#)
- 539. [ifx_get_char](#)
- 540. [ifx_getsqlca](#)
- 541. [ifx_htmltbl_result](#)
- 542. [ifx_nullformat](#)
- 543. [ifx_num_fields](#)
- 544. [ifx_num_rows](#)
- 545. [ifx_pconnect](#)
- 546. [ifx_prepare](#)
- 547. [ifx_query](#)
- 548. [ifx_textasvarchar](#)
- 549. [ifx_update_blob](#)
- 550. [ifx_update_char](#)
- 551. [ifxus_close_slob](#)
- 552. [ifxus_create_slob](#)
- 553. [ifxus_open_slob](#)
- 554. [ifxus_read_slob](#)
- 555. [ifxus_seek_slob](#)
- 556. [ifxus_tell_slob](#)
- 557. [ifxus_write_slob](#)
- 558. [ignore_user_abort](#)
- 559. [imagearc](#)
- 560. [imagechar](#)
- 561. [imagecharup](#)
- 562. [imagecolorallocate](#)
- 563. [imagecolorat](#)
- 564. [imagecolorclosest](#)
- 565. [imagecolordeallocate](#)
- 566. [imagecolorexact](#)
- 567. [imagecolorresolve](#)
- 568. [imagecolorset](#)
- 569. [imagecolorsforindex](#)
- 570. [imagecolorstotal](#)
- 571. [imagecolortransparent](#)
- 572. [imagecopy](#)
- 573. [imagecopyresized](#)
- 574. [imagecreate](#)
- 575. [imagecreatefromgif](#)
- 576. [imagecreatefromjpeg](#)

- 577. [imagecreatefrompng](#)
- 578. [imagecreatefromstring](#)
- 579. [imagecreatefromwbmp](#)
- 580. [imagedashedline](#)
- 581. [imagedestroy](#)
- 582. [imagefill](#)
- 583. [imagefilledpolygon](#)
- 584. [imagefilledrectangle](#)
- 585. [imagefilltoborder](#)
- 586. [imagefontheight](#)
- 587. [imagefontwidth](#)
- 588. [imagegammacorrect](#)
- 589. [imagegif](#)
- 590. [imageinterlace](#)
- 591. [imagejpeg](#)
- 592. [imageline](#)
- 593. [imageloadfont](#)
- 594. [imagepng](#)
- 595. [imagepolygon](#)
- 596. [imagepsbbox](#)
- 597. [imagepsencodefont](#)
- 598. [imagepsextendfont](#)
- 599. [imagepsfreefont](#)
- 600. [imagepsloadfont](#)
- 601. [imagepsslantfont](#)
- 602. [imagepstext](#)
- 603. [imagerectangle](#)
- 604. [imagesetpixel](#)
- 605. [imagestring](#)
- 606. [imagestringup](#)
- 607. [imagesx](#)
- 608. [imagesy](#)
- 609. [imagettfbbox](#)
- 610. [imagettftext](#)
- 611. [imagetypes](#)
- 612. [imagewbmp](#)
- 613. [imap_8bit](#)
- 614. [imap_alerts](#)
- 615. [imap_append](#)
- 616. [imap_base64](#)
- 617. [imap_binary](#)
- 618. [imap_body](#)
- 619. [imap_check](#)
- 620. [imap_clearflag_full](#)
- 621. [imap_close](#)
- 622. [imap_createmailbox](#)
- 623. [imap_delete](#)
- 624. [imap_deletemailbox](#)
- 625. [imap_errors](#)
- 626. [imap_expunge](#)
- 627. [imap_fetch_overview](#)

- 628. [imap_fetchbody](#)
- 629. [imap_fetchheader](#)
- 630. [imap_fetchstructure](#)
- 631. [imap_getmailboxes](#)
- 632. [imap_getsubscribed](#)
- 633. [imap_header](#)
- 634. [imap_headerinfo](#)
- 635. [imap_headers](#)
- 636. [imap_last_error](#)
- 637. [imap_listmailbox](#)
- 638. [imap_listsubscribed](#)
- 639. [imap_mail](#)
- 640. [imap_mail_compose](#)
- 641. [imap_mail_copy](#)
- 642. [imap_mail_move](#)
- 643. [imap_mailboxmsginfo](#)
- 644. [imap_mime_header_decode](#)
- 645. [imap_msgno](#)
- 646. [imap_num_msg](#)
- 647. [imap_num_recent](#)
- 648. [imap_open](#)
- 649. [imap_ping](#)
- 650. [imap_qprint](#)
- 651. [imap_renamemailbox](#)
- 652. [imap_reopen](#)
- 653. [imap_rfc822_parse_adrlist](#)
- 654. [imap_rfc822_parse_headers](#)
- 655. [imap_rfc822_write_address](#)
- 656. [imap_scanmailbox](#)
- 657. [imap_search](#)
- 658. [imap_setflag_full](#)
- 659. [imap_sort](#)
- 660. [imap_status](#)
- 661. [imap_subscribe](#)
- 662. [imap_uid](#)
- 663. [imap_undelete](#)
- 664. [imap_unsubscribe](#)
- 665. [imap_utf7_decode](#)
- 666. [imap_utf7_encode](#)
- 667. [imap_utf8](#)
- 668. [implode](#)
- 669. [in_array](#)
- 670. [include\(\)](#)
- 671. [include_once\(\)](#)
- 672. [ingres_autocommit](#)
- 673. [ingres_close](#)
- 674. [ingres_commit](#)
- 675. [ingres_connect](#)
- 676. [ingres_fetch_array](#)
- 677. [ingres_fetch_object](#)
- 678. [ingres_fetch_row](#)

- 679. [ingres_field_length](#)
- 680. [ingres_field_name](#)
- 681. [ingres_field_nullable](#)
- 682. [ingres_field_precision](#)
- 683. [ingres_field_scale](#)
- 684. [ingres_field_type](#)
- 685. [ingres_num_fields](#)
- 686. [ingres_num_rows](#)
- 687. [ingres_pconnect](#)
- 688. [ingres_query](#)
- 689. [ingres_rollback](#)
- 690. [ini_alter](#)
- 691. [ini_get](#)
- 692. [ini_restore](#)
- 693. [ini_set](#)
- 694. [intval](#)
- 695. [ip2long](#)
- 696. [iptcpase](#)
- 697. [is_array](#)
- 698. [is_bool](#)
- 699. [is_dir](#)
- 700. [is_double](#)
- 701. [is_executable](#)
- 702. [is_file](#)
- 703. [is_float](#)
- 704. [is_int](#)
- 705. [is_integer](#)
- 706. [is_link](#)
- 707. [is_long](#)
- 708. [is_numeric](#)
- 709. [is_object](#)
- 710. [is_readable](#)
- 711. [is_real](#)
- 712. [is_resource](#)
- 713. [is_string](#)
- 714. [is_subclass_of](#)
- 715. [is_uploaded_file](#)
- 716. [is_writable](#)
- 717. [isset](#)

J

- 718. [jddayofweek](#)
- 719. [jdmonthname](#)
- 720. [jdtofrance](#)
- 721. [jdtogregorian](#)
- 722. [jdtojewish](#)
- 723. [jdtojulian](#)
- 724. [jdtonix](#)
- 725. [jewishtojd](#)

- 726. [join](#)
- 727. [juliantojd](#)

K

- 728. [key](#)
- 729. [krsort](#)
- 730. [ksort](#)

L

- 731. [lcg_value](#)
- 732. [ldap_add](#)
- 733. [ldap_bind](#)
- 734. [ldap_close](#)
- 735. [ldap_compare](#)
- 736. [ldap_connect](#)
- 737. [ldap_count_entries](#)
- 738. [ldap_delete](#)
- 739. [ldap_dn2ufn](#)
- 740. [ldap_err2str](#)
- 741. [ldap_errno](#)
- 742. [ldap_error](#)
- 743. [ldap_explode_dn](#)
- 744. [ldap_first_attribute](#)
- 745. [ldap_first_entry](#)
- 746. [ldap_free_result](#)
- 747. [ldap_get_attributes](#)
- 748. [ldap_get_dn](#)
- 749. [ldap_get_entries](#)
- 750. [ldap_get_option](#)
- 751. [ldap_get_values](#)
- 752. [ldap_get_values_len](#)
- 753. [ldap_list](#)
- 754. [ldap_mod_add](#)
- 755. [ldap_mod_del](#)
- 756. [ldap_mod_replace](#)
- 757. [ldap_modify](#)
- 758. [ldap_next_attribute](#)
- 759. [ldap_next_entry](#)
- 760. [ldap_read](#)
- 761. [ldap_search](#)
- 762. [ldap_set_option](#)
- 763. [ldap_unbind](#)
- 764. [leak](#)
- 765. [levenshtein](#)
- 766. [link](#)
- 767. [linkinfo](#)
- 768. [list](#)
- 769. [listen](#)

- 770. [localeconv](#)
- 771. [localtime](#)
- 772. [log](#)
- 773. [log10](#)
- 774. [long2ip](#)
- 775. [lstat](#)
- 776. [ltrim](#)

M

- 777. [mail](#)
- 778. [max](#)
- 779. [mcaldump](#)
- 780. [mcaldump](#)
- 781. [mcaldump](#)
- 782. [mcaldump](#)
- 783. [mcaldump](#)
- 784. [mcaldump](#)
- 785. [mcaldump](#)
- 786. [mcaldump](#)
- 787. [mcaldump](#)
- 788. [mcaldump](#)
- 789. [mcaldump](#)
- 790. [mcaldump](#)
- 791. [mcaldump](#)
- 792. [mcaldump](#)
- 793. [mcaldump](#)
- 794. [mcaldump](#)
- 795. [mcaldump](#)
- 796. [mcaldump](#)
- 797. [mcaldump](#)
- 798. [mcaldump](#)
- 799. [mcaldump](#)
- 800. [mcaldump](#)
- 801. [mcaldump](#)
- 802. [mcaldump](#)
- 803. [mcaldump](#)
- 804. [mcaldump](#)
- 805. [mcaldump](#)
- 806. [mcaldump](#)
- 807. [mcaldump](#)
- 808. [mcaldump](#)
- 809. [mcaldump](#)
- 810. [mcaldump](#)
- 811. [mcaldump](#)
- 812. [mcaldump](#)
- 813. [mcaldump](#)
- 814. [mcaldump](#)
- 815. [mcaldump](#)
- 816. [mcaldump](#)

- 817. [mcrypt_time_valid](#)
- 818. [mcrypt_cbc](#)
- 819. [mcrypt_cfb](#)
- 820. [mcrypt_create_iv](#)
- 821. [mcrypt_decrypt](#)
- 822. [mcrypt_ecb](#)
- 823. [mcrypt_enc_get_algorithms_name](#)
- 824. [mcrypt_enc_get_block_size](#)
- 825. [mcrypt_enc_get_iv_size](#)
- 826. [mcrypt_enc_get_key_size](#)
- 827. [mcrypt_enc_get_modes_name](#)
- 828. [mcrypt_enc_get_supported_key_sizes](#)
- 829. [mcrypt_enc_is_block_algorithm](#)
- 830. [mcrypt_enc_is_block_algorithm_mode](#)
- 831. [mcrypt_enc_is_block_mode](#)
- 832. [mcrypt_enc_self_test](#)
- 833. [mcrypt_encrypt](#)
- 834. [mcrypt_generic](#)
- 835. [mcrypt_generic_end](#)
- 836. [mcrypt_generic_init](#)
- 837. [mcrypt_get_block_size](#)
- 838. [mcrypt_get_cipher_name](#)
- 839. [mcrypt_get_iv_size](#)
- 840. [mcrypt_get_key_size](#)
- 841. [mcrypt_list_algorithms](#)
- 842. [mcrypt_list_modes](#)
- 843. [mcrypt_module_get_algo_block_size](#)
- 844. [mcrypt_module_get_algo_key_size](#)
- 845. [mcrypt_module_get_algo_supported_key_sizes](#)
- 846. [mcrypt_module_is_block_algorithm](#)
- 847. [mcrypt_module_is_block_algorithm_mode](#)
- 848. [mcrypt_module_is_block_mode](#)
- 849. [mcrypt_module_open](#)
- 850. [mcrypt_module_self_test](#)
- 851. [mcrypt_ofb](#)
- 852. [md5](#)
- 853. [mdecrypt_generic](#)
- 854. [metaphone](#)
- 855. [method_exists](#)
- 856. [mhash](#)
- 857. [mhash_count](#)
- 858. [mhash_get_block_size](#)
- 859. [mhash_get_hash_name](#)
- 860. [mhash_keygen_s2k](#)
- 861. [microtime](#)
- 862. [min](#)
- 863. [mkdir](#)
- 864. [mktime](#)
- 865. [move_uploaded_file](#)
- 866. [mysql](#)
- 867. [mysql_affected_rows](#)

- 868. [mysql_close](#)
- 869. [mysql_connect](#)
- 870. [mysql_create_db](#)
- 871. [mysql_createdb](#)
- 872. [mysql_data_seek](#)
- 873. [mysql_dbname](#)
- 874. [mysql_drop_db](#)
- 875. [mysql_dropdb](#)
- 876. [mysql_error](#)
- 877. [mysql_fetch_array](#)
- 878. [mysql_fetch_field](#)
- 879. [mysql_fetch_object](#)
- 880. [mysql_fetch_row](#)
- 881. [mysql_field_seek](#)
- 882. [mysql_fieldflags](#)
- 883. [mysql_fieldlen](#)
- 884. [mysql_fieldname](#)
- 885. [mysql_fieldtable](#)
- 886. [mysql_fieldtype](#)
- 887. [mysql_free_result](#)
- 888. [mysql_freeresult](#)
- 889. [mysql_list_dbs](#)
- 890. [mysql_list_fields](#)
- 891. [mysql_list_tables](#)
- 892. [mysql_listdbs](#)
- 893. [mysql_listfields](#)
- 894. [mysql_listtables](#)
- 895. [mysql_num_fields](#)
- 896. [mysql_num_rows](#)
- 897. [mysql_numfields](#)
- 898. [mysql_numrows](#)
- 899. [mysql_pconnect](#)
- 900. [mysql_query](#)
- 901. [mysql_regcase](#)
- 902. [mysql_result](#)
- 903. [mysql_select_db](#)
- 904. [mysql_selectdb](#)
- 905. [mysql_tablename](#)
- 906. [mssql_close](#)
- 907. [mssql_connect](#)
- 908. [mssql_data_seek](#)
- 909. [mssql_fetch_array](#)
- 910. [mssql_fetch_field](#)
- 911. [mssql_fetch_object](#)
- 912. [mssql_fetch_row](#)
- 913. [mssql_field_length](#)
- 914. [mssql_field_name](#)
- 915. [mssql_field_seek](#)
- 916. [mssql_field_type](#)
- 917. [mssql_free_result](#)
- 918. [mssql_get_last_message](#)

- 919. [mssql_min_error_severity](#)
- 920. [mssql_min_message_severity](#)
- 921. [mssql_num_fields](#)
- 922. [mssql_num_rows](#)
- 923. [mssql_pconnect](#)
- 924. [mssql_query](#)
- 925. [mssql_result](#)
- 926. [mssql_select_db](#)
- 927. [mt_getrandmax](#)
- 928. [mt_rand](#)
- 929. [mt_srand](#)
- 930. [mysql_affected_rows](#)
- 931. [mysql_change_user](#)
- 932. [mysql_close](#)
- 933. [mysql_connect](#)
- 934. [mysql_create_db](#)
- 935. [mysql_data_seek](#)
- 936. [mysql_db_name](#)
- 937. [mysql_db_query](#)
- 938. [mysql_drop_db](#)
- 939. [mysql_errno](#)
- 940. [mysql_error](#)
- 941. [mysql_fetch_array](#)
- 942. [mysql_fetch_assoc](#)
- 943. [mysql_fetch_field](#)
- 944. [mysql_fetch_lengths](#)
- 945. [mysql_fetch_object](#)
- 946. [mysql_fetch_row](#)
- 947. [mysql_field_flags](#)
- 948. [mysql_field_len](#)
- 949. [mysql_field_name](#)
- 950. [mysql_field_seek](#)
- 951. [mysql_field_table](#)
- 952. [mysql_field_type](#)
- 953. [mysql_free_result](#)
- 954. [mysql_insert_id](#)
- 955. [mysql_list_dbs](#)
- 956. [mysql_list_fields](#)
- 957. [mysql_list_tables](#)
- 958. [mysql_num_fields](#)
- 959. [mysql_num_rows](#)
- 960. [mysql_pconnect](#)
- 961. [mysql_query](#)
- 962. [mysql_result](#)
- 963. [mysql_select_db](#)
- 964. [mysql_tablename](#)

N

- 965. [natcasesort](#)

- 966. [natsort](#)
- 967. [next](#)
- 968. [nl2br](#)
- 969. [number_format](#)

O

- 970. [ob_end_clean](#)
- 971. [ob_end_flush](#)
- 972. [ob_get_contents](#)
- 973. [ob_get_length](#)
- 974. [ob_implicit_flush](#)
- 975. [ob_start](#)
- 976. [ocibindbyname](#)
- 977. [ocicolumnisnull](#)
- 978. [ocicolumnname](#)
- 979. [ocicolumnsize](#)
- 980. [ocicolumntype](#)
- 981. [ocicommit](#)
- 982. [ocidefinebyname](#)
- 983. [ocierror](#)
- 984. [ociexecute](#)
- 985. [ocifetch](#)
- 986. [ocifetchinto](#)
- 987. [ocifetchstatement](#)
- 988. [ocifreecursor](#)
- 989. [ocifreedesc](#)
- 990. [ocifreestatement](#)
- 991. [ociinternaldebug](#)
- 992. [ocilogout](#)
- 993. [ocilogon](#)
- 994. [ocinewcursor](#)
- 995. [ocinewdescriptor](#)
- 996. [ocinlogon](#)
- 997. [ocinumcols](#)
- 998. [ociparse](#)
- 999. [ociplogon](#)
- 1000. [ocireult](#)
- 1001. [ocirollback](#)
- 1002. [ocirowcount](#)
- 1003. [ociserverversion](#)
- 1004. [ocistatementtype](#)
- 1005. [octdec](#)
- 1006. [odbc_autocommit](#)
- 1007. [odbc_binmode](#)
- 1008. [odbc_close](#)
- 1009. [odbc_close_all](#)
- 1010. [odbc_columnprivileges](#)
- 1011. [odbc_columns](#)
- 1012. [odbc_commit](#)

- 1013. [odbc_connect](#)
- 1014. [odbc_cursor](#)
- 1015. [odbc_do](#)
- 1016. [odbc_error](#)
- 1017. [odbc_errormsg](#)
- 1018. [odbc_exec](#)
- 1019. [odbc_execute](#)
- 1020. [odbc_fetch_into](#)
- 1021. [odbc_fetch_row](#)
- 1022. [odbc_field_len](#)
- 1023. [odbc_field_name](#)
- 1024. [odbc_field_num](#)
- 1025. [odbc_field_precision](#)
- 1026. [odbc_field_scale](#)
- 1027. [odbc_field_type](#)
- 1028. [odbc_foreignkeys](#)
- 1029. [odbc_free_result](#)
- 1030. [odbc_gettypeinfo](#)
- 1031. [odbc_longreadlen](#)
- 1032. [odbc_num_fields](#)
- 1033. [odbc_num_rows](#)
- 1034. [odbc_pconnect](#)
- 1035. [odbc_prepare](#)
- 1036. [odbc_primarykeys](#)
- 1037. [odbc_procedurecolumns](#)
- 1038. [odbc_procedures](#)
- 1039. [odbc_result](#)
- 1040. [odbc_result_all](#)
- 1041. [odbc_rollback](#)
- 1042. [odbc_setoption](#)
- 1043. [odbc_specialcolumns](#)
- 1044. [odbc_statistics](#)
- 1045. [odbc_tableprivileges](#)
- 1046. [odbc_tables](#)
- 1047. [opendir](#)
- 1048. [openlog](#)
- 1049. [openssl_free_key](#)
- 1050. [openssl_get_privatekey](#)
- 1051. [openssl_get_publickey](#)
- 1052. [openssl_open](#)
- 1053. [openssl_seal](#)
- 1054. [openssl_sign](#)
- 1055. [openssl_verify](#)
- 1056. [ora_bind](#)
- 1057. [ora_close](#)
- 1058. [ora_columnname](#)
- 1059. [ora_columnsize](#)
- 1060. [ora_columntype](#)
- 1061. [ora_commit](#)
- 1062. [ora_commitoff](#)
- 1063. [ora_commiton](#)

- 1064. [ora_do](#)
- 1065. [ora_error](#)
- 1066. [ora_errorcode](#)
- 1067. [ora_exec](#)
- 1068. [ora_fetch](#)
- 1069. [ora_fetch_into](#)
- 1070. [ora_getcolumn](#)
- 1071. [ora_logoff](#)
- 1072. [ora_logon](#)
- 1073. [ora_numcols](#)
- 1074. [ora_numrows](#)
- 1075. [ora_open](#)
- 1076. [ora_parse](#)
- 1077. [ora_plogon](#)
- 1078. [ora_rollback](#)
- 1079. [ord](#)
- 1080. [ovrimos_close](#)
- 1081. [ovrimos_close_all](#)
- 1082. [ovrimos_commit](#)
- 1083. [ovrimos_connect](#)
- 1084. [ovrimos_cursor](#)
- 1085. [ovrimos_exec](#)
- 1086. [ovrimos_execute](#)
- 1087. [ovrimos_fetch_into](#)
- 1088. [ovrimos_fetch_row](#)
- 1089. [ovrimos_field_len](#)
- 1090. [ovrimos_field_name](#)
- 1091. [ovrimos_field_num](#)
- 1092. [ovrimos_field_type](#)
- 1093. [ovrimos_free_result](#)
- 1094. [ovrimos_longreadlen](#)
- 1095. [ovrimos_num_fields](#)
- 1096. [ovrimos_num_rows](#)
- 1097. [ovrimos_prepare](#)
- 1098. [ovrimos_result](#)
- 1099. [ovrimos_result_all](#)
- 1100. [ovrimos_rollback](#)

P

- 1101. [pack](#)
- 1102. [parse_str](#)
- 1103. [parse_url](#)
- 1104. [passthru](#)
- 1105. [pclose](#)
- 1106. [pdf_add_annotation](#)
- 1107. [pdf_add_outline](#)
- 1108. [pdf_arc](#)
- 1109. [pdf_begin_page](#)
- 1110. [pdf_circle](#)

- 1111. [pdf_clip](#)
- 1112. [pdf_close](#)
- 1113. [pdf_close_image](#)
- 1114. [pdf_closepath](#)
- 1115. [pdf_closepath_fill_stroke](#)
- 1116. [pdf_closepath_stroke](#)
- 1117. [pdf_continue_text](#)
- 1118. [pdf_curveto](#)
- 1119. [pdf_end_page](#)
- 1120. [pdf_endpath](#)
- 1121. [pdf_execute_image](#)
- 1122. [pdf_fill](#)
- 1123. [pdf_fill_stroke](#)
- 1124. [pdf_get_image_height](#)
- 1125. [pdf_get_image_width](#)
- 1126. [pdf_get_parameter](#)
- 1127. [pdf_get_value](#)
- 1128. [pdf_lineto](#)
- 1129. [pdf_moveto](#)
- 1130. [pdf_open](#)
- 1131. [pdf_open_gif](#)
- 1132. [pdf_open_image_file](#)
- 1133. [pdf_open_jpeg](#)
- 1134. [pdf_open_memory_image](#)
- 1135. [pdf_open_png](#)
- 1136. [pdf_open_tiff](#)
- 1137. [pdf_place_image](#)
- 1138. [pdf_put_image](#)
- 1139. [pdf_rect](#)
- 1140. [pdf_restore](#)
- 1141. [pdf_rotate](#)
- 1142. [pdf_save](#)
- 1143. [pdf_scale](#)
- 1144. [pdf_set_border_color](#)
- 1145. [pdf_set_border_dash](#)
- 1146. [pdf_set_border_style](#)
- 1147. [pdf_set_char_spacing](#)
- 1148. [pdf_set_duration](#)
- 1149. [pdf_set_font](#)
- 1150. [pdf_set_horiz_scaling](#)
- 1151. [pdf_set_info](#)
- 1152. [pdf_set_leading](#)
- 1153. [pdf_set_parameter](#)
- 1154. [pdf_set_text_matrix](#)
- 1155. [pdf_set_text_pos](#)
- 1156. [pdf_set_text_rendering](#)
- 1157. [pdf_set_text_rise](#)
- 1158. [pdf_set_transition](#)
- 1159. [pdf_set_value](#)
- 1160. [pdf_set_word_spacing](#)
- 1161. [pdf_setdash](#)

- 1162. [pdf_setflat](#)
- 1163. [pdf_setgray](#)
- 1164. [pdf_setgray_fill](#)
- 1165. [pdf_setgray_stroke](#)
- 1166. [pdf_setlinecap](#)
- 1167. [pdf_setlinejoin](#)
- 1168. [pdf_setlinewidth](#)
- 1169. [pdf_setmiterlimit](#)
- 1170. [pdf_setrgbcolor](#)
- 1171. [pdf_setrgbcolor_fill](#)
- 1172. [pdf_setrgbcolor_stroke](#)
- 1173. [pdf_show](#)
- 1174. [pdf_show_boxed](#)
- 1175. [pdf_show_xy](#)
- 1176. [pdf_skew](#)
- 1177. [pdf_stringwidth](#)
- 1178. [pdf_stroke](#)
- 1179. [pdf_translate](#)
- 1180. [pfpro_cleanup](#)
- 1181. [pfpro_init](#)
- 1182. [pfpro_process](#)
- 1183. [pfpro_process_raw](#)
- 1184. [pfpro_version](#)
- 1185. [pfsockopen](#)
- 1186. [pg_client_encoding](#)
- 1187. [pg_close](#)
- 1188. [pg_cmdtuples](#)
- 1189. [pg_connect](#)
- 1190. [pg_dbname](#)
- 1191. [pg_end_copy](#)
- 1192. [pg_errormessage](#)
- 1193. [pg_exec](#)
- 1194. [pg_fetch_array](#)
- 1195. [pg_fetch_object](#)
- 1196. [pg_fetch_row](#)
- 1197. [pg_fieldisnull](#)
- 1198. [pg_fieldname](#)
- 1199. [pg_fieldnum](#)
- 1200. [pg_fieldprtlen](#)
- 1201. [pg_fieldsize](#)
- 1202. [pg_fieldtype](#)
- 1203. [pg_freeresult](#)
- 1204. [pg_getlastoid](#)
- 1205. [pg_host](#)
- 1206. [pg_loclose](#)
- 1207. [pg_locreate](#)
- 1208. [pg_loexport](#)
- 1209. [pg_loimport](#)
- 1210. [pg_loopen](#)
- 1211. [pg_loread](#)
- 1212. [pg_loreadall](#)

- 1213. [pg_lounlink](#)
- 1214. [pg_lowrite](#)
- 1215. [pg_numfields](#)
- 1216. [pg_numrows](#)
- 1217. [pg_options](#)
- 1218. [pg_pconnect](#)
- 1219. [pg_port](#)
- 1220. [pg_put_line](#)
- 1221. [pg_result](#)
- 1222. [pg_set_client_encoding](#)
- 1223. [pg_trace](#)
- 1224. [pg_tty](#)
- 1225. [pg_untrace](#)
- 1226. [php_logo_guid](#)
- 1227. [php_sapi_name](#)
- 1228. [php_uname](#)
- 1229. [phpcredits](#)
- 1230. [phpinfo](#)
- 1231. [phpversion](#)
- 1232. [pi](#)
- 1233. [popen](#)
- 1234. [pos](#)
- 1235. [posix_ctermid](#)
- 1236. [posix_getcwd](#)
- 1237. [posix_getegid](#)
- 1238. [posix_geteuid](#)
- 1239. [posix_getgid](#)
- 1240. [posix_getgrgid](#)
- 1241. [posix_getgrnam](#)
- 1242. [posix_getgroups](#)
- 1243. [posix_getlogin](#)
- 1244. [posix_getpgid](#)
- 1245. [posix_getpgrp](#)
- 1246. [posix_getpid](#)
- 1247. [posix_getppid](#)
- 1248. [posix_getpwnam](#)
- 1249. [posix_getpwuid](#)
- 1250. [posix_getrlimit](#)
- 1251. [posix_getsid](#)
- 1252. [posix_getuid](#)
- 1253. [posix_isatty](#)
- 1254. [posix_kill](#)
- 1255. [posix_mkfifo](#)
- 1256. [posix_setgid](#)
- 1257. [posix_setpgid](#)
- 1258. [posix_setsid](#)
- 1259. [posix_setuid](#)
- 1260. [posix_times](#)
- 1261. [posix_ttyname](#)
- 1262. [posix_uname](#)
- 1263. [pow](#)

- 1264. [preg_grep](#)
- 1265. [preg_match](#)
- 1266. [preg_match_all](#)
- 1267. [preg_quote](#)
- 1268. [preg_replace](#)
- 1269. [preg_split](#)
- 1270. [prev](#)
- 1271. [print](#)
- 1272. [print_r](#)
- 1273. [printf](#)
- 1274. [pspell_add_to_personal](#)
- 1275. [pspell_add_to_session](#)
- 1276. [pspell_check](#)
- 1277. [pspell_clear_session](#)
- 1278. [pspell_config_create](#)
- 1279. [pspell_config_ignore](#)
- 1280. [pspell_config_mode](#)
- 1281. [pspell_config_personal](#)
- 1282. [pspell_config_repl](#)
- 1283. [pspell_config_runtogether](#)
- 1284. [pspell_config_save_repl](#)
- 1285. [pspell_new](#)
- 1286. [pspell_new_config](#)
- 1287. [pspell_new_personal](#)
- 1288. [pspell_save_wordlist](#)
- 1289. [pspell_store_replacement](#)
- 1290. [pspell_suggest](#)
- 1291. [putenv](#)

Q

- 1292. [quoted_printable_decode](#)
- 1293. [quotemeta](#)

R

- 1294. [rad2deg](#)
- 1295. [rand](#)
- 1296. [range](#)
- 1297. [rawurldecode](#)
- 1298. [rawurlencode](#)
- 1299. [read](#)
- 1300. [read_exif_data](#)
- 1301. [readdir](#)
- 1302. [readfile](#)
- 1303. [readgzfile](#)
- 1304. [readline](#)
- 1305. [readline_add_history](#)
- 1306. [readline_clear_history](#)
- 1307. [readline_completion_function](#)

- 1308. [readline_info](#)
- 1309. [readline_list_history](#)
- 1310. [readline_read_history](#)
- 1311. [readline_write_history](#)
- 1312. [readlink](#)
- 1313. [realpath](#)
- 1314. [recode](#)
- 1315. [recode_file](#)
- 1316. [recode_string](#)
- 1317. [register_shutdown_function](#)
- 1318. [rename](#)
- 1319. [require\(\)](#)
- 1320. [require_once\(\)](#)
- 1321. [reset](#)
- 1322. [restore_error_handler](#)
- 1323. [rewind](#)
- 1324. [rewinddir](#)
- 1325. [rmdir](#)
- 1326. [round](#)
- 1327. [rsort](#)
- 1328. [rtrim](#)

S

- 1329. [satellite_caught_exception](#)
- 1330. [satellite_exception_id](#)
- 1331. [satellite_exception_value](#)
- 1332. [sem_acquire](#)
- 1333. [sem_get](#)
- 1334. [sem_release](#)
- 1335. [serialize](#)
- 1336. [sesam_affected_rows](#)
- 1337. [sesam_commit](#)
- 1338. [sesam_connect](#)
- 1339. [sesam_diagnostic](#)
- 1340. [sesam_disconnect](#)
- 1341. [sesam_errormsg](#)
- 1342. [sesam_execimm](#)
- 1343. [sesam_fetch_array](#)
- 1344. [sesam_fetch_result](#)
- 1345. [sesam_fetch_row](#)
- 1346. [sesam_field_array](#)
- 1347. [sesam_field_name](#)
- 1348. [sesam_free_result](#)
- 1349. [sesam_num_fields](#)
- 1350. [sesam_query](#)
- 1351. [sesam_rollback](#)
- 1352. [sesam_seek_row](#)
- 1353. [sesam_settransaction](#)
- 1354. [session_cache_limiter](#)

- 1355. [session_decode](#)
- 1356. [session_destroy](#)
- 1357. [session_encode](#)
- 1358. [session_get_cookie_params](#)
- 1359. [session_id](#)
- 1360. [session_is_registered](#)
- 1361. [session_module_name](#)
- 1362. [session_name](#)
- 1363. [session_register](#)
- 1364. [session_save_path](#)
- 1365. [session_set_cookie_params](#)
- 1366. [session_set_save_handler](#)
- 1367. [session_start](#)
- 1368. [session_unregister](#)
- 1369. [session_unset](#)
- 1370. [set_error_handler](#)
- 1371. [set_file_buffer](#)
- 1372. [set_magic_quotes_runtime](#)
- 1373. [set_time_limit](#)
- 1374. [setcookie](#)
- 1375. [setlocale](#)
- 1376. [settype](#)
- 1377. [shm_attach](#)
- 1378. [shm_close](#)
- 1379. [shm_delete](#)
- 1380. [shm_detach](#)
- 1381. [shm_get_var](#)
- 1382. [shm_open](#)
- 1383. [shm_put_var](#)
- 1384. [shm_read](#)
- 1385. [shm_remove](#)
- 1386. [shm_remove_var](#)
- 1387. [shm_size](#)
- 1388. [shm_write](#)
- 1389. [show_source](#)
- 1390. [shuffle](#)
- 1391. [similar_text](#)
- 1392. [sin](#)
- 1393. [sizeof](#)
- 1394. [sleep](#)
- 1395. [snmp_get_quick_print](#)
- 1396. [snmp_set_quick_print](#)
- 1397. [snmpget](#)
- 1398. [snmpset](#)
- 1399. [snmpwalk](#)
- 1400. [snmpwalkoid](#)
- 1401. [socket](#)
- 1402. [socket_get_status](#)
- 1403. [socket_set_blocking](#)
- 1404. [socket_set_timeout](#)
- 1405. [sort](#)

- 1406. [soundex](#)
- 1407. [split](#)
- 1408. [spliti](#)
- 1409. [sprintf](#)
- 1410. [sql_regcase](#)
- 1411. [sqrt](#)
- 1412. [srand](#)
- 1413. [sscanf](#)
- 1414. [stat](#)
- 1415. [str_pad](#)
- 1416. [str_repeat](#)
- 1417. [str_replace](#)
- 1418. [strcasecmp](#)
- 1419. [strchr](#)
- 1420. [strcmp](#)
- 1421. [strcoll](#)
- 1422. [strcspn](#)
- 1423. [strerror](#)
- 1424. [strftime](#)
- 1425. [strip_tags](#)
- 1426. [stripslashes](#)
- 1427. [stripslashes](#)
- 1428. [stristr](#)
- 1429. [strlen](#)
- 1430. [strnatcasecmp](#)
- 1431. [strnatcmp](#)
- 1432. [strncasecmp](#)
- 1433. [strncmp](#)
- 1434. [strpos](#)
- 1435. [strrchr](#)
- 1436. [strrev](#)
- 1437. [strrpos](#)
- 1438. [strspn](#)
- 1439. [strstr](#)
- 1440. [strtok](#)
- 1441. [strtolower](#)
- 1442. [strtotime](#)
- 1443. [strtoupper](#)
- 1444. [strtr](#)
- 1445. [strval](#)
- 1446. [substr](#)
- 1447. [substr_count](#)
- 1448. [substr_replace](#)
- 1449. [swf_actiongeturl](#)
- 1450. [swf_actiongotoframe](#)
- 1451. [swf_actiongotolabel](#)
- 1452. [swf_actionnextframe](#)
- 1453. [swf_actionplay](#)
- 1454. [swf_actionprevframe](#)
- 1455. [swf_actionsettarget](#)
- 1456. [swf_actionstop](#)

- 1457. [swf_actiontogglequality](#)
- 1458. [swf_actionwaitforframe](#)
- 1459. [swf_addbuttonrecord](#)
- 1460. [swf_addcolor](#)
- 1461. [swf_closefile](#)
- 1462. [swf_definebitmap](#)
- 1463. [swf_definefont](#)
- 1464. [swf_defineline](#)
- 1465. [swf_definepoly](#)
- 1466. [swf_definerect](#)
- 1467. [swf_definetext](#)
- 1468. [swf_endbutton](#)
- 1469. [swf_enddoaction](#)
- 1470. [swf_endshape](#)
- 1471. [swf_endsymbol](#)
- 1472. [swf_fontsize](#)
- 1473. [swf_fontslant](#)
- 1474. [swf_fonttracking](#)
- 1475. [swf_getbitmapinfo](#)
- 1476. [swf_getfontinfo](#)
- 1477. [swf_getframe](#)
- 1478. [swf_labelframe](#)
- 1479. [swf_lookat](#)
- 1480. [swf_modifyobject](#)
- 1481. [swf_mulcolor](#)
- 1482. [swf_nextid](#)
- 1483. [swf_oncondition](#)
- 1484. [swf_openfile](#)
- 1485. [swf_ortho](#)
- 1486. [swf_ortho2](#)
- 1487. [swf_perspective](#)
- 1488. [swf_placeobject](#)
- 1489. [swf_polarview](#)
- 1490. [swf_popmatrix](#)
- 1491. [swf_posround](#)
- 1492. [swf_pushmatrix](#)
- 1493. [swf_removeobject](#)
- 1494. [swf_rotate](#)
- 1495. [swf_scale](#)
- 1496. [swf_setfont](#)
- 1497. [swf_setframe](#)
- 1498. [swf_shapearc](#)
- 1499. [swf_shapecurveto](#)
- 1500. [swf_shapecurveto3](#)
- 1501. [swf_shapefillbitmapclip](#)
- 1502. [swf_shapefillbitmaptile](#)
- 1503. [swf_shapefilloff](#)
- 1504. [swf_shapefillsolid](#)
- 1505. [swf_shapelinesolid](#)
- 1506. [swf_shapelineto](#)
- 1507. [swf_shapemoveto](#)

- 1508. [swf_showframe](#)
- 1509. [swf_startbutton](#)
- 1510. [swf_startdoaction](#)
- 1511. [swf_startshape](#)
- 1512. [swf_startsymbol](#)
- 1513. [swf_textwidth](#)
- 1514. [swf_translate](#)
- 1515. [swf_viewport](#)
- 1516. [sybase_affected_rows](#)
- 1517. [sybase_close](#)
- 1518. [sybase_connect](#)
- 1519. [sybase_data_seek](#)
- 1520. [sybase_fetch_array](#)
- 1521. [sybase_fetch_field](#)
- 1522. [sybase_fetch_object](#)
- 1523. [sybase_fetch_row](#)
- 1524. [sybase_field_seek](#)
- 1525. [sybase_free_result](#)
- 1526. [sybase_get_last_message](#)
- 1527. [sybase_min_client_severity](#)
- 1528. [sybase_min_error_severity](#)
- 1529. [sybase_min_message_severity](#)
- 1530. [sybase_min_server_severity](#)
- 1531. [sybase_num_fields](#)
- 1532. [sybase_num_rows](#)
- 1533. [sybase_pconnect](#)
- 1534. [sybase_query](#)
- 1535. [sybase_result](#)
- 1536. [sybase_select_db](#)
- 1537. [symlink](#)
- 1538. [syslog](#)
- 1539. [system](#)

T

- 1540. [tan](#)
- 1541. [tempnam](#)
- 1542. [textdomain](#)
- 1543. [time](#)
- 1544. [tmpfile](#)
- 1545. [touch](#)
- 1546. [trigger_error](#)
- 1547. [trim](#)

U

- 1548. [uasort](#)
- 1549. [ucfirst](#)
- 1550. [ucwords](#)
- 1551. [uksort](#)

- 1552. [umask](#)
- 1553. [uniqid](#)
- 1554. [unixtojd](#)
- 1555. [unlink](#)
- 1556. [unpack](#)
- 1557. [unserialize](#)
- 1558. [unset](#)
- 1559. [urldecode](#)
- 1560. [urlencode](#)
- 1561. [user_error](#)
- 1562. [usleep](#)
- 1563. [usort](#)
- 1564. [utf8_decode](#)
- 1565. [utf8_encode](#)

V

- 1566. [var_dump](#)
- 1567. [virtual](#)

W

- 1568. [wddx_add_vars](#)
- 1569. [wddx_deserialize](#)
- 1570. [wddx_packet_end](#)
- 1571. [wddx_packet_start](#)
- 1572. [wddx_serialize_value](#)
- 1573. [wddx_serialize_vars](#)
- 1574. [wordwrap](#)
- 1575. [write](#)

X

- 1576. [xml_error_string](#)
- 1577. [xml_get_current_byte_index](#)
- 1578. [xml_get_current_column_number](#)
- 1579. [xml_get_current_line_number](#)
- 1580. [xml_get_error_code](#)
- 1581. [xml_parse](#)
- 1582. [xml_parse_into_struct](#)
- 1583. [xml_parser_create](#)
- 1584. [xml_parser_free](#)
- 1585. [xml_parser_get_option](#)
- 1586. [xml_parser_set_option](#)
- 1587. [xml_set_character_data_handler](#)
- 1588. [xml_set_default_handler](#)
- 1589. [xml_set_element_handler](#)
- 1590. [xml_set_external_entity_ref_handler](#)
- 1591. [xml_set_notation_decl_handler](#)

- 1592. [xml_set_object](#)
- 1593. [xml_set_processing_instruction_handler](#)
- 1594. [xml_set_unparsed_entity_decl_handler](#)
- 1595. [xmldoc](#)
- 1596. [xmldocfile](#)
- 1597. [xmldata](#)
- 1598. [xslt_closelog](#)
- 1599. [xslt_create](#)
- 1600. [xslt_errno](#)
- 1601. [xslt_error](#)
- 1602. [xslt_fetch_result](#)
- 1603. [xslt_free](#)
- 1604. [xslt_openlog](#)
- 1605. [xslt_output_begintransform](#)
- 1606. [xslt_output_endtransform](#)
- 1607. [xslt_process](#)
- 1608. [xslt_run](#)
- 1609. [xslt_set_sax_handler](#)
- 1610. [xslt_transform](#)

Y

- 1611. [yaz_addinfo](#)
- 1612. [yaz_close](#)
- 1613. [yaz_connect](#)
- 1614. [yaz_errno](#)
- 1615. [yaz_error](#)
- 1616. [yaz_hits](#)
- 1617. [yaz_range](#)
- 1618. [yaz_record](#)
- 1619. [yaz_search](#)
- 1620. [yaz_syntax](#)
- 1621. [yaz_wait](#)
- 1622. [yp_first](#)
- 1623. [yp_get_default_domain](#)
- 1624. [yp_master](#)
- 1625. [yp_match](#)
- 1626. [yp_next](#)
- 1627. [yp_order](#)

Z

- 1628. [zend_logo_guid](#)

Index des concepts

[\[Notes en ligne\]](#)

1629. , , , ,

A

- 1630. [Accède aux objets CORBA](#)
- 1631. [Accepte une connexion sur une socket](#)
- 1632. [Active l'approximation des translation d'objets.](#)
- 1633. [Active l'option décidant si, lors de la déconnexion du client, le script doit poursuivre son exécution ou non.](#)
- 1634. [Active la sauvegarde des paires de remplacement](#)
- 1635. [Active la validation automatique](#)
- 1636. [Active le debugger interne de PHP](#)
- 1637. [Active le suivi d'une connexion PostgreSQL](#)
- 1638. [Active ou désactive l'affichage des données de debuggage](#)
- 1639. [Active ou désactive l'entrelacement](#)
- 1640. [Active ou désactive le mode autocommit](#)
- 1641. [Active ou désactive le mode passif](#)
- 1642. [Active/désactive l'envoi implicite](#)
- 1643. [Active/désactive l'option magic_quotes_runtime.](#)
- 1644. [Active/désactive le mode bloquant d'une socket](#)
- 1645. [Additionne deux nombres de taille arbitraire](#)
- 1646. [Addition de 2 nombres GMP](#)
- 1647. [Affecte et/ou retourne l'identifiant de session courante](#)
- 1648. [Affecte et/ou retourne le chemin de sauvegarde de la session courante](#)
- 1649. [Affecte et/ou retourne le module courant de session courante](#)
- 1650. [Affecte et/ou retourne le nom de la session courante](#)
- 1651. [Affecte le gestionnaire par défaut](#)
- 1652. [Affecte les gestionnaires d'entité non déclaré.](#)
- 1653. [Affecte les gestionnaires d'instructions exécutables.](#)
- 1654. [Affecte les gestionnaires de caractère bruts](#)
- 1655. [Affecte les gestionnaires de début et de fin](#)
- 1656. [Affecte les gestionnaires de notation](#)
- 1657. [Affecte les options d'un analyseur XML](#)
- 1658. [Affecte un type à une variable](#)
- 1659. [Affecte une nouvelle date de modification à un fichier.](#)
- 1660. [Affiche de nombreuses informations sur le PHP](#)
- 1661. [Affiche des informations lisibles pour une variable.](#)
- 1662. [Affiche hw document](#)
- 1663. [Affiche la partie du fichier située après le pointeur du fichier.](#)
- 1664. [Affiche le frame courant](#)
- 1665. [Affiche le frame nommé.](#)
- 1666. [Affiche le résultat sous la forme d'une table HTML.](#)
- 1667. [Affiche ou affecte le paramètre "apache request notes".](#)
- 1668. [Affiche un fichier](#)
- 1669. [Affiche un fichier compressé](#)

- 1670. [Affiche un message et termine le script courant](#)
- 1671. [Affiche un résultat sous forme de table HTML](#)
- 1672. [Affiche un texte à la position courante](#)
- 1673. [Affiche un texte à une position](#)
- 1674. [Affiche un texte à une position donnée](#)
- 1675. [Affiche un texte dans un rectangle](#)
- 1676. [Affiche un texte sur une nouvelle ligne](#)
- 1677. [Affiche une chaîne](#)
- 1678. [Affiche une chaîne formatée](#)
- 1679. [Affiche une image WBMP](#)
- 1680. [Affiche une ou plusieurs chaînes](#)
- 1681. [Ajoute des slashes dans une chaîne, comme en langage C.](#)
- 1682. [Ajoute le mot au dictionnaire personnel](#)
- 1683. [Ajoute le mot au dictionnaire personnel de la session courante](#)
- 1684. [Ajoute un antislash devant tous les caractères méta](#)
- 1685. [Ajoute un attribut](#)
- 1686. [Ajoute un attribut et une valeur à la structure globale](#)
- 1687. [Ajoute un enregistrement dans une base dBase](#)
- 1688. [Ajoute un signet à la page courante](#)
- 1689. [Ajoute un signet sur la page courante](#)
- 1690. [Ajoute un slash devant tous les caractères spéciaux](#)
- 1691. [Ajoute une annotation, Ajoute une annotation](#)
- 1692. [Ajoute une césure à une chaîne tous les n caractères](#)
- 1693. [Ajoute une chaîne dans une boîte aux lettres.](#)
- 1694. [Ajoute une entrée à un dossier LDAP](#)
- 1695. [Ajoute une ligne à l'historique](#)
- 1696. [Ajouter des variables à un paquet WDDX](#)
- 1697. [Alias de `odbc_field_len\(\)`](#)
- 1698. [Aligne les dessins sur le chemin courant](#)
- 1699. [Aligne sur le chemin courant](#)
- 1700. [Alloue une couleur pour une image](#)
- 1701. [Analyse un bloc binaire IPTC <http://www.iptc.org/> et recherche les balises simples.](#)
- 1702. [Analyse un entête mail](#)
- 1703. [Analyse un fichier en fonction d'un format](#)
- 1704. [Analyse une chaîne d'adresse](#)
- 1705. [Analyse une chaîne, et en déduit des variables et leur valeur.](#)
- 1706. [Analyse une fonction en fonction d'un format](#)
- 1707. [Analyse une requête](#)
- 1708. [Analyse une requête SQL](#)
- 1709. [Analyse une structure XML](#)
- 1710. [Analyse une URL et retourne ses composants](#)
- 1711. [Analyse, exécute et lit une requête](#)
- 1712. [Annule les transactions en cours](#)
- 1713. [Annule un bit](#)
- 1714. [Annule une transaction, Annule une transaction, Annule une transaction, Annule une transaction, Annule une transaction](#)
- 1715. [Annule une transaction SESAM](#)
- 1716. [Appelle une fonction utilisateur](#)
- 1717. [Appelle une méthode d'un composant](#)
- 1718. [Appelle une méthode utilisateur d'un objet](#)
- 1719. [Applique une correction gamma à l'image](#)

- 1720. [Applique une feuille de style à un fichier](#)
- 1721. [arc cosinus](#)
- 1722. [arc sinus](#)
- 1723. [arc tangent](#)
- 1724. [arc tangent de deux variables](#)
- 1725. [Arrête l'animation flash.](#)
- 1726. [Arrondi](#)
- 1727. [Arrondi à l'entier inférieur](#)
- 1728. [arrondi au nombre supérieur](#)
- 1729. [Attend une connexion sur une socket](#)
- 1730. [Attributs d'un objet](#)
- 1731. [Attributs de l'objet dans l'ancrage](#)
- 1732. [Attributs des ancrages d'un document](#)
- 1733. [Attributs des documents fils d'un groupe](#)
- 1734. [Attributs des parents](#)
- 1735. [Authentification d'une connexion FTP](#)
- 1736. [Avance d'un frame](#)
- 1737. [Avance le pointeur interne d'un tableau](#)

B

- 1738. [Balise de corps d'un document](#)

C

- 1739. [Calcule des statistiques sur une table](#)
- 1740. [Calcule l'intersection de tableaux](#)
- 1741. [Calcule la clé metaphone d'une chaîne](#)
- 1742. [Calcule la différence entre deux tableaux](#)
- 1743. [Calcule la distance Levenshtein entre deux chaînes](#)
- 1744. [Calcule la similarité de deux chaînes.](#)
- 1745. [Calcule la valeur de hash demandée par EZMLM](#)
- 1746. [Calcule la valeur soundex d'une chaîne](#)
- 1747. [Calcule le nombre de couleur d'une palette](#)
- 1748. [Calcule le polynôme crc32 d'une chaîne](#)
- 1749. [Calcule un hash](#)
- 1750. [Calcule un md5 avec la chaîne](#)
- 1751. [Carré parfait GMP](#)
- 1752. [Change de dossier](#)
- 1753. [Change de dossier, et passe au dossier parent](#)
- 1754. [Change l'espacement des caractères](#)
- 1755. [Change l'inclinaison de la police courante](#)
- 1756. [Change la couleur dans une palette à l'index donné.](#)
- 1757. [Change la police courante](#)
- 1758. [Change la position courante](#)
- 1759. [Change la taille de la police](#)
- 1760. [Change la valeur d'une option de configuration, Change la valeur d'une option de configuration](#)
- 1761. [Change le "umask" courant](#)
- 1762. [Change le codage vectoriel d'un caractère dans une police.](#)
- 1763. [Change le dossier courant](#)

- 1764. [Change le groupe possesseur du fichier](#)
- 1765. [Change le groupe propriétaire du fichier](#)
- 1766. [Change le mode de suggestion](#)
- 1767. [Change le mode du fichier](#)
- 1768. [Change le nom de session de l'utilisateur actif.](#)
- 1769. [Change les informations locales](#)
- 1770. [Charge un fichier ouvert sur un serveur FTP](#)
- 1771. [Charge un fichier sur un serveur FTP](#)
- 1772. [Charge un nouveau dictionnaire](#), [Charge un nouveau dictionnaire](#), [Charge un nouveau dictionnaire](#)
- 1773. [Charge un nouveau dictionnaire avec un dictionnaire personnel](#)
- 1774. [Charge une extension PHP à la volée](#)
- 1775. [Charge une nouvelle police](#)
- 1776. [Charge une police PostScript Type 1 depuis un fichier](#)
- 1777. [Chemin du dossier courant](#)
- 1778. [Choisi l'encodage du client](#)
- 1779. [Choisi la couleur grise comme couleur de remplissage](#)
- 1780. [Choisi la couleur grise comme couleur de remplissage et de dessin.](#)
- 1781. [Choisi la couleur rgb comme couleur de dessin et de remplissage](#)
- 1782. [Choisi la couleur rgb comme couleur de remplissage](#), [Choisi la couleur rgb comme couleur de remplissage](#)
- 1783. [Choisi la durée de transition entre deux pages](#)
- 1784. [Choisi la rotation](#)
- 1785. [Choisi le fichier qui contient le dictionnaire personnel](#)
- 1786. [Choisi le fichier qui contient les paires de remplacement](#)
- 1787. [Choisi le mode de remplissage par texture répétée.](#)
- 1788. [Choisi le mode de remplissage par texture.](#)
- 1789. [Choisi le mode par défaut des objets BLOB pour toutes les requêtes SELECT.](#)
- 1790. [Choisi le mode par défaut des objets BYTE](#)
- 1791. [Choisi le mode par défaut des objets text](#)
- 1792. [Choisi le niveau de qualité haut ou bas.](#)
- 1793. [Choisi le paramètre linecap](#)
- 1794. [Choisi une fonction utilisateur comme gestionnaire d'erreurs](#)
- 1795. [Choisit un niveau de gris comme couleur de dessin](#)
- 1796. [Choisit une couleur rgb comme couleur de dessin](#)
- 1797. [Choisit une couleur rgb comme couleur de dessin et de remplissage.](#)
- 1798. [Choisit une couleur rgb comme couleur de remplissage](#)
- 1799. [Classe dossier](#)
- 1800. [Clos un paquet WDDX](#)
- 1801. [Colorisation d'une chaîne](#)
- 1802. [Colorisation de la syntaxe d'un fichier](#), [Colorisation de la syntaxe d'un fichier](#)
- 1803. [Combine plusieurs tableaux ensemble, récursivement](#)
- 1804. [Commence l'analyse d'un fichier XML](#)
- 1805. [Commence la définition d'un bouton](#)
- 1806. [Commence la description d'une liste d'action pour la frame courante.](#)
- 1807. [Commence la transformation XSLT](#)
- 1808. [Commence un nouveau chemin](#)
- 1809. [Commence une forme complexe](#)
- 1810. [Commence une nouvelle page](#), [Commence une nouvelle page](#)
- 1811. [Commencer un nouveau paquet WDDX avec une structure](#)
- 1812. [Compacte des données dans une chaîne binaire](#)
- 1813. [Compacte une base dBase](#)

- 1814. [Comparaison binaire de chaînes](#)
- 1815. [Comparaison binaire de chaînes, insensible à la casse.](#)
- 1816. [Comparaison binaire des premiers caractères](#)
- 1817. [Comparaison binaires de chaînes de caractères](#)
- 1818. [Comparaison de chaînes localisée](#)
- 1819. [Comparaisons de chaîne par ordre "naturel"](#)
- 1820. [Comparaisons de chaîne par ordre "naturel" insensible à la casse](#)
- 1821. [Compare des nombres GMP](#)
- 1822. [Compare deux dates](#)
- 1823. [Compare deux nombres de taille arbitraire](#)
- 1824. [Compare les valeurs des attributs trouvés dans un ND](#)
- 1825. [Complète la définition de la forme courante.](#)
- 1826. [Complète un tableau jusqu'à la longueur spécifiée, avec une valeur.](#)
- 1827. [Complète une chaîne avec une autre](#)
- 1828. [Compresse une chane avec Gz](#)
- 1829. [Compte de population](#)
- 1830. [Compte le nombre d'élément d'un tableau](#)
- 1831. [Compte le nombre d'entrées d'une recherche](#)
- 1832. [Compte le nombre de champs d'une base dBase.](#)
- 1833. [Compte le nombre de ligne déjà lues dans un résultat.](#)
- 1834. [Compte le nombre de sous chaînes](#)
- 1835. [Compte le nombre de valeurs dans un tableau](#)
- 1836. [Compter le nombre d'enregistrements dans une base dBase.](#)
- 1837. [Connection persistante à un serveur Oracle](#)
- 1838. [Connexion à un serveur](#)
- 1839. [Connexion à une source](#)
- 1840. [Considère deux mots accolés comme un composé.](#)
- 1841. [Contenu d'un document](#)
- 1842. [Contrôle la situation, l'aparance et la zone active du bouton courant.](#)
- 1843. [Convertie des données 8bit en texte UTF-7.](#)
- 1844. [Convertit d'octal en décimal](#)
- 1845. [Convertit de binaire en décimal](#)
- 1846. [Convertit de décimal en binaire](#)
- 1847. [Convertit de décimal en hexadécimal](#)
- 1848. [Convertit de décimal en octal](#)
- 1849. [Convertit de hexadécimal en décimal](#)
- 1850. [Convertit du texte en UTF8](#)
- 1851. [Convertit la chaîne d'un alphabet cyrillique vers un autre.](#)
- 1852. [Convertit le nombre de jour du calendrier Julien en date du calendrier Julien.](#)
- 1853. [Convertit le nombre de jours du calendrier julien en date du calendrier français républicain](#)
- 1854. [Convertit le nombre de jours du calendrier Julien en date du calendrier Julien.](#)
- 1855. [Convertit le nombre de jours du calendrier julien en date du calendrier juif.](#)
- 1856. [Convertit le nombre de jours du calendrier Julien en date grégorienne.](#)
- 1857. [Convertit les attributs d'un objet en tableau](#)
- 1858. [Convertit les nouvelles lignes en HTML \(<BR\)](#)
- 1859. [Convertit tous les caractères spéciaux en équivalent HTML.](#), [Convertit tous les caractères spéciaux en équivalent HTML.](#)
- 1860. [Convertit un ND dans un format plus accessible](#)
- 1861. [Convertit un nombre de jour Julien en timestamp UNIX](#)
- 1862. [Convertit un nombre en degré en radians](#)
- 1863. [Convertit un nombre en des bases arbitraires](#)

1864. [Convertit un nombre en radian en degrés](#)
1865. [Convertit un nombre GMP en chaîne](#)
1866. [Convertit un nombre GMP en entier](#)
1867. [Convertit un numéro d'erreur LDAP en message d'erreur.](#)
1868. [Convertit un tableau en un objet](#)
1869. [Convertit un texte Hébreux logique en texte visual](#)
1870. [Convertit un texte hébreux logique en texte visuel avec les nouvelles lignes de conversion.](#)
1871. [Convertit un timestamp UNIX en nombre de jours Julien](#)
1872. [Convertit un une chaîne ISO-8859-1 en UTF-8](#)
1873. [Convertit un une chaîne UTF-8 en ISO-8859](#)
1874. [Convertit une adresse IP \(IPv4\) en adresse IP numérique](#)
1875. [Convertit une chaîne à 8 bits en une chaîne à base64.](#)
1876. [Convertit une chaîne à 8 bits en une chaîne à guillemets.](#)
1877. [Convertit une chaîne à guillemets en une chaîne à 8 bits.](#)
1878. [Convertit une chaîne contenant une adresse \(IPv4\) IP numérique en adresse littérale.](#)
1879. [Convertit une date du calendrier français républicain en nombre de jours du calendrier julien.](#)
1880. [Convertit une date du calendrier juif en nombre de jours du calendrier julien.](#)
1881. [Convertit une date grégorienne en nombre de jours du calendrier julien.](#)
1882. [Convertit une valeur binaire en hexadécimal](#)
1883. [Copie des objets](#)
1884. [Copie et redimensionne une partie d'une image](#)
1885. [Copie les messages spécifiés dans une boîte aux lettres.](#)
1886. [Copie un fichier](#)
1887. [Copie une partie d'une image](#)
1888. [cosinus](#)
1889. [Cré un nouveau analyseur XSL](#)
1890. [Création d'un analyseur XML](#)
1891. [Crée ou ouvre un bloc de mémoire partagée](#)
1892. [Crée ou ouvre un segment de mémoire partagée.](#)
1893. [Crée un arbre d'objet PHP, à partir d'un document XML.](#)
1894. [Crée un dossier, Crée un dossier](#)
1895. [Crée un fichier avec un nom unique](#)
1896. [Crée un fichier fifo \(first in, first out\) \(un pipe nommé\)](#)
1897. [Crée un fichier temporaire](#)
1898. [Crée un lien](#)
1899. [Crée un lien symbolique](#)
1900. [Crée un message MIME](#)
1901. [Crée un nombre GMP](#)
1902. [Crée un nouveau calendrier](#)
1903. [Crée un nouveau document](#)
1904. [Crée un nouveau document FDF](#)
1905. [Crée un objet BLOB](#)
1906. [Crée un objet char](#)
1907. [Crée un objet de grande taille](#)
1908. [Crée un objet DOM à partir d'un fichier XML](#)
1909. [Crée un objet DOM pour un document XML](#)
1910. [Crée un objet SLOB et l'ouvre](#)
1911. [Crée un processus de pointeur de fichier](#)
1912. [Crée un tableau](#)
1913. [Crée un tableau contenant les variables et leur valeur](#)
1914. [Crée un tableau contenant un intervalle d'entiers](#)

- 1915. [Crée un vecteur d'initialisation à partir d'une source aléatoire.](#)
- 1916. [Crée une base de données dBase](#)
- 1917. [Crée une base de données mSQL, Crée une base de données mSQL](#)
- 1918. [Crée une base de données MySQL](#)
- 1919. [Crée une configuration utilisée pour ouvrir un dictionnaire](#)
- 1920. [Crée une connexion persistante](#)
- 1921. [Crée une fonction anonyme \(style lambda\)](#)
- 1922. [Crée une image à partir d'une chaîne](#)
- 1923. [Crée une image depuis un fichier WBMP](#)
- 1924. [Crée une nouvelle boîte aux lettres](#)
- 1925. [Crée une nouvelle image](#)
- 1926. [Crée une nouvelle image à partir d'un fichier ou d'une URL.](#)
- 1927. [Crée une nouvelle image *JPEG* à partir d'un fichier ou d'une URL](#)
- 1928. [Crée une nouvelle image *PNG* à partir d'un fichier ou d'une URL](#)
- 1929. [Crée une nouvelle référence sur un composant COM](#)
- 1930. [Crée une socket \(point de communication\)](#)
- 1931. [Crée une variable PHP à partir d'une valeur linéarisée](#)

D

- 1932. [Déclenche une erreur utilisateur](#)
- 1933. [Décode les éléments MIME d'un entête](#)
- 1934. [Décode les données de session à partir d'une chaîne](#)
- 1935. [Décode un texte encodé en BASE64](#)
- 1936. [Décode une chaîne](#)
- 1937. [Décode une chaîne en MIME base64](#)
- 1938. [Décode une chaîne encodée URL](#)
- 1939. [Décode une chaîne modifiée UTF-7.](#)
- 1940. [Décode une chaîne URL](#)
- 1941. [Décompresse une chaîne gz-compressée](#)
- 1942. [Déconditionne des données depuis une chaîne binaire.](#)
- 1943. [Déconnecte d'un serveur LDAP](#)
- 1944. [Déconnection d'un serveur Oracle](#)
- 1945. [Déconnexion d'une base SESAM](#)
- 1946. [Décrit la librairie dbm utilisée](#)
- 1947. [Décrit une transition utilisée pour déclencher une liste d'actions.](#)
- 1948. [Décrypte](#)
- 1949. [Décrypte un texte](#)
- 1950. [Dédoublonne un tableau](#)
- 1951. [Définit le point de vue de l'utilisateur en coordonnées polaire.](#)
- 1952. [Définit la couleur transparente](#)
- 1953. [Définit les fonctions utilisateurs de stockage des sessions](#)
- 1954. [Définit un polygone.](#)
- 1955. [Définit un rectangle](#)
- 1956. [Définit un symbole](#)
- 1957. [Définit une chaîne de texte](#)
- 1958. [Définit une constante](#)
- 1959. [Définit une image bitmap](#)
- 1960. [Définit une ligne](#)
- 1961. [Définit une police.](#)

1962. [Définit une projection orthogonale à 2 dimensions entre les coordonnées utilisateur et le port courant.](#)
1963. [Définit une projection orthogonale à 3 dimensions entre les coordonnées utilisateur et le port courant](#)
1964. [Définit une projection orthogonale entre les coordonnées utilisateur et le port courant.](#)
1965. [Définit une transformation de vue](#)
1966. [Démarre une section de texte](#)
1967. [Dépile la matrice de transformation.](#)
1968. [Dépile un élément au début d'un tableau](#)
1969. [Dépile un élément de la fin d'un tableau](#)
1970. [Déplace le pointeur courant dans un fichier compressé](#)
1971. [Déplace le pointeur interne](#)
1972. [Déplace le pointeur interne de ligne](#)
1973. [Déplace le pointeur interne de lignes](#)
1974. [Déplace le pointeur interne de résultat](#)
1975. [Déplace les messages spécifiés dans une boîte aux lettres.](#)
1976. [Déplace un curseur à défilement](#)
1977. [Déplace un fichier téléchargé](#)
1978. [Déplace un objet](#)
1979. [Déquote une chaîne quotée avec addcslashes](#)
1980. [Désallouune une couleur pour une image](#)
1981. [Détermine le nombre de décimales par défaut pour les fonctions de précision mathématiques.](#)
1982. [Détermine si un objet est une sous-classe](#)
1983. [Détermine si une extension est chargée ou non.](#)
1984. [Détermine si une variable est affectée, Détermine si une variable est affectée](#)
1985. [Détermine si une variable est de type double](#)
1986. [Détermine si une variable est de type float](#)
1987. [Détermine si une variable est de type int](#)
1988. [Détermine si une variable est de type integer, Détermine si une variable est de type integer](#)
1989. [Détermine si une variable est de type object](#)
1990. [Détermine si une variable est de type real](#)
1991. [Détermine si une variable est de type string](#)
1992. [Détermine si une variable est un tableau](#)
1993. [Détermine si une variable est un tableau booléen](#)
1994. [Détermine si une variable est un type numérique](#)
1995. [Détermine si une variable est une ressource](#)
1996. [Déterminez le rendu du texte](#)
1997. [Détruit es données du buffer de sortie, et éteind la bufferisation de sortie](#)
1998. [Détruit toutes les données enregistrées d'une session](#)
1999. [Détruit toutes les variables de session](#)
2000. [Détruit un analyseur XML](#)
2001. [Détruit un analyseur XSLT](#)
2002. [Détruit un bloc de mémoire partagée](#)
2003. [Détruit un document](#)
2004. [détruit une image](#)
2005. [Détruit une variable](#)
2006. [Déverrouille un objet](#)
2007. [Defface un enregistrement dans une base dBase](#)
2008. [Demande d'informations d'arbre sur une entité du réseau.](#)
2009. [Dessine le long du chemin](#)
2010. [Dessine un arc](#)
2011. [Dessine un arc de cercle](#)
2012. [Dessine un caractère horizontalement](#)

- 2013. [Dessine un caractère verticalement](#)
- 2014. [Dessine un cercle](#), [Dessine un cercle](#)
- 2015. [Dessine un pixel](#)
- 2016. [Dessine un polygone](#)
- 2017. [Dessine un polygone rempli](#)
- 2018. [Dessine un rectangle](#), [Dessine un rectangle](#), [Dessine un rectangle](#)
- 2019. [Dessine un rectangle rempli](#)
- 2020. [Dessine un texte avec une police TrueType](#)
- 2021. [Dessine un texte sur une image avec une police PostScript Type1](#)
- 2022. [Dessine une arc de cercle](#)
- 2023. [Dessine une chaîne horizontale](#)
- 2024. [Dessine une chaîne verticale](#)
- 2025. [Dessine une courbe](#), [Dessine une courbe](#)
- 2026. [Dessine une courbe Bézier cubique](#)
- 2027. [Dessine une courbe de Bézier quadratique entre deux points.](#)
- 2028. [Dessine une ellipse partielle](#)
- 2029. [Dessine une ligne](#), [Dessine une ligne](#), [Dessine une ligne](#)
- 2030. [Dessine une ligne le long du chemin](#)
- 2031. [Dessine une ligne pointillée](#)
- 2032. [Dessine une ligne, relativement](#)
- 2033. [Determine le rendu du texte](#)
- 2034. [Determine si un pointeur de fichier est un terminal interactif](#)
- 2035. [Distance de Hamming](#)
- 2036. [Divise deux nombres de taille arbitraire](#)
- 2037. [Divise deux nombres GMP](#), [Divise deux nombres GMP](#)
- 2038. [Division exacte de nombres GMP](#)
- 2039. [Divisions de 2 nombres GMP](#)
- 2040. [Draw a line](#)
- 2041. [Dumpe les informations d'une variable.](#)
- 2042. [Duplique un objet BLOB](#)

E

- 2043. [Echappe les méta-caractères Shell](#)
- 2044. [Echappe une chaîne de caractères pour qu'elle soit utilisée en ligne de commande.](#)
- 2045. [Echappement des caractères spéciaux des expressions régulières.](#)
- 2046. [Eclatement d'une chaîne par expression régulière.](#)
- 2047. [Ecrire dans un bloc de mémoire partagée](#)
- 2048. [Ecrit dans l'historique](#)
- 2049. [Ecrit dans un fichier](#)
- 2050. [Ecrit dans un fichier compressé](#)
- 2051. [Ecrit la valeur courante de l'option quick_print de la librairie UCD.](#)
- 2052. [Ecrit sur une socket](#)
- 2053. [Ecrit un document PDF dans un fichier](#)
- 2054. [Ecrit un fichier compressé en mode binaire](#)
- 2055. [Ecrit un objet de grande taille](#)
- 2056. [Ecrit une chaîne dans un objet SLOB](#)
- 2057. [Ecriture du fichier en mode binaire](#)
- 2058. [Efface des objets](#)
- 2059. [Efface et remplace une portion de tableau](#)

- 2060. [Efface l'historique](#)
- 2061. [Efface le cache de la fonction "stat"](#)
- 2062. [Efface le fichier d'historique](#)
- 2063. [Efface le résultat de la mémoire](#)
- 2064. [Efface les espaces de fin de chaîne](#)
- 2065. [Efface tous les messages marqués pour l'effacement.](#)
- 2066. [Efface un événement dans un calendrier MCAL.](#)
- 2067. [Efface un événement dans un agenda ICAP](#)
- 2068. [Efface un attribut](#)
- 2069. [Efface un calendrier](#)
- 2070. [Efface un dossier](#), [Efface un dossier](#)
- 2071. [Efface un fichier](#)
- 2072. [Efface un fichier sur un serveur FTP](#)
- 2073. [Efface un objet de grande taille](#)
- 2074. [Efface une base de données mSQL](#), [Efface une base de données mSQL](#)
- 2075. [Efface une base de données MySQL](#)
- 2076. [Efface une boîte aux lettres](#)
- 2077. [Efface une entrée](#)
- 2078. [Efface une entrée dans un dossier](#)
- 2079. [Efface une valeur](#)
- 2080. [Efface une variable de la mémoire partagée.](#)
- 2081. [Effacer](#)
- 2082. [Effectue une requête partielle pour l'URI spécifiée et renvoie toutes les informations.](#)
- 2083. [Effectue une rotation](#)
- 2084. [Effectue une sous-requête Apache](#)
- 2085. [Effectue une transaction avec Payflow Pro](#)
- 2086. [Elève un nombre à la puissance n-ième.](#)
- 2087. [Empile la matrice de transformation courante dans la pile.](#)
- 2088. [Empile un ou plusieurs éléments à la fin d'un tableau](#)
- 2089. [Empile un ou plusieurs éléments au début d'un tableau](#)
- 2090. [Enclenche la bufferisation de sortie](#)
- 2091. [Encode les données de session dans une chaîne](#)
- 2092. [Encode une chaîne en MIME base64](#)
- 2093. [Encode une chaîne en URL](#)
- 2094. [Encode une chaîne en URL, selon la RFC1738](#)
- 2095. [Encrypte](#)
- 2096. [Encrypte un texte](#)
- 2097. [Encrypte une chaîne avec un DES](#)
- 2098. [Encrypte/décrypte des données en mode CBC](#)
- 2099. [Encrypte/décrypte des données en mode CFB](#)
- 2100. [Encrypte/décrypte des données en mode ECB](#)
- 2101. [Encrypte/décrypte des données en mode OFB](#)
- 2102. [Enlève la marque d'effacement d'un message.](#)
- 2103. [Enlève les balises HTML et PHP](#)
- 2104. [Enlève les espaces de début de chaîne.](#)
- 2105. [Enlève les espaces de fin de chaîne.](#)
- 2106. [Enlève les espaces de fin et de fin de chaîne.](#)
- 2107. [Enlève les slash ajoutés par la fonction addslashes](#)
- 2108. [Enlève un objet](#)
- 2109. [Enregistre l'environnement courant](#)
- 2110. [Enregistre un événement dans un agenda ICAP](#)

- 2111. [Enregistre un nouvel événement dans un calendrier MCAL.](#)
- 2112. [Enregistre une fonction de complétion](#)
- 2113. [Enregistre une fonction pour exécution à l'extinction](#)
- 2114. [Enregistre une image dans un fichier PDF pour utilisation ultérieure.](#)
- 2115. [Enregistre une paire de remplacement pour un mot](#)
- 2116. [Enregistre une variable dans la session courante](#)
- 2117. [Enregistrer plusieurs valeurs dans un paquet WDDX](#)
- 2118. [Enregistrer une valeur dans un paquet WDDX](#)
- 2119. [Envoi de mail](#)
- 2120. [Envoi tout le contenu généré dans un fichier](#)
- 2121. [Envoie la commande SITE au serveur](#)
- 2122. [Envoie le document PDF dans un buffer mémoire](#)
- 2123. [Envoie les données du buffer de sortie, et éteint la bufferisation de sortie](#)
- 2124. [Envoie un cookie](#)
- 2125. [Envoie un entête HTTP](#)
- 2126. [Envoie un message d'erreur quelque part](#)
- 2127. [Envoie un message mail](#)
- 2128. [Envoie un objet SNMP](#)
- 2129. [Envoie un signal à un process](#)
- 2130. [Envoie une chaîne au serveur PostgreSQL](#)
- 2131. [Envoie une image GIF vers un navigateur ou un fichier](#)
- 2132. [Envoie une image JPEG vers un navigateur ou un fichier.](#)
- 2133. [Envoie une image PNG vers un navigateur ou un fichier.](#)
- 2134. [Envoie une requête à une base Sybase](#)
- 2135. [Envoie une requête Informix](#)
- 2136. [Envoie une requête mSQL](#)
- 2137. [Envoie une requête MySQL à un serveur MySQL](#)
- 2138. [Envoie une requête SQL](#)
- 2139. [Envoie une requête SQL à un serveur Ingres II](#)
- 2140. [Envoie une requête SQL à un serveur MySQL](#)
- 2141. [Envoie une transaction brute à Payflow Pro](#)
- 2142. [ET logique](#)
- 2143. [Etablit une connexion à un serveur Oracle](#)
- 2144. [Etablit une connexion persistante.](#)
- 2145. [Eteind l'alarme d'un événement, Eteind l'alarme d'un événement](#)
- 2146. [Eteind la librairie Payflow Pro](#)
- 2147. [Etend ou condense une police de caractères](#)
- 2148. [Evalue une chaîne comme un script PHP](#)
- 2149. [Exécute immédiatement une requête SQL](#)
- 2150. [Exécute un programme externe](#)
- 2151. [Exécute un programme externe et affiche le résultat brut.](#)
- 2152. [Exécute un programme externe et affiche le résultat.](#)
- 2153. [Exécute une commande](#)
- 2154. [Exécute une commande analysée sur un pointeur Oracle.](#)
- 2155. [Exécute une fonction sur chacun des membres d'un tableau.](#)
- 2156. [Exécute une requête, Exécute une requête](#)
- 2157. [Exécute une requête mSQL](#)
- 2158. [Exécute une requête préparée, Exécute une requête préparée](#)
- 2159. [Exécute une requête SESAM](#)
- 2160. [Exécute une requête SQL déjà préparée.](#)
- 2161. [Exécute une requête SQL préparée.](#)

- 2162. [Exécute une requête sur une base Interbase](#)
- 2163. [Exécute une requête SQL](#)
- 2164. [Exécute une transformation XSLT](#)
- 2165. [Exécute une session CURL](#)
- 2166. [exponentielle](#)
- 2167. [Exporte un objet de grande vers un fichier](#)
- 2168. [Expression régulière globale](#)
- 2169. [Expression régulière standard](#), [Expression régulière standard](#)
- 2170. [Extrait toutes les balises meta d'un fichier, et les retourne sous forme d'un tableau.](#)
- 2171. [Extrait une clé publique d'un certificat](#)
- 2172. [Extrait une portion de tableau](#)

F

- 2173. [Factorielle GMP](#)
- 2174. [Fait du processus courant un chef de session](#)
- 2175. [Fait une liste détaillée de fichier dans un dossier.](#)
- 2176. [Ferme et clos le chemin](#)
- 2177. [Ferme la connexion à l'historique système](#)
- 2178. [Ferme la connexion Hyperwave](#)
- 2179. [Ferme la connexion MySQL](#)
- 2180. [Ferme le chemin](#)
- 2181. [Ferme le chemin courant](#)
- 2182. [Ferme le chemin et dessine le long du chemin](#)
- 2183. [Ferme le fichier courant Shockwave Flash](#)
- 2184. [Ferme le fichier et dessine une ligne le long du chemin.](#)
- 2185. [Ferme le flot ICAP](#)
- 2186. [Ferme le pointeur sur le dossier](#)
- 2187. [Ferme toutes les connexions aux serveur ovrinos](#)
- 2188. [Ferme toutes les connexions ODBC](#)
- 2189. [Ferme un bloc de mémoire partagée](#)
- 2190. [Ferme un document PDF](#)
- 2191. [Ferme un fichier](#)
- 2192. [Ferme un fichier PDF](#)
- 2193. [Ferme un objet de grande taille](#)
- 2194. [Ferme un objet SLOB](#)
- 2195. [Ferme un pointeur Oracle](#)
- 2196. [Ferme un pointeur sur un fichier compressé](#)
- 2197. [Ferme un processus de pointeur de fichier](#)
- 2198. [Ferme une base](#)
- 2199. [Ferme une base dBase](#)
- 2200. [Ferme une base de données dbm](#)
- 2201. [Ferme une connexion](#)
- 2202. [Ferme une connexion à un serveur Informix](#)
- 2203. [Ferme une connexion à un serveur Ingres](#)
- 2204. [Ferme une connexion à une base de données Interbase.](#)
- 2205. [Ferme une connexion FTP](#)
- 2206. [Ferme une connexion MCAL](#)
- 2207. [Ferme une connexion MS SQL Server](#)
- 2208. [Ferme une connexion mSQL](#)

- 2209. [Ferme une connexion ODBC](#)
- 2210. [Ferme une connexion Oracle](#)
- 2211. [Ferme une connexion Sybase](#)
- 2212. [Ferme une connexion YAZ](#)
- 2213. [Ferme une document FDF](#)
- 2214. [Ferme une image](#)
- 2215. [Ferme une session CURL](#)
- 2216. [Ferme une socket](#)
- 2217. [Fixe d'offset d'un champs](#)
- 2218. [Fixe et lit différentes options d'assertions](#)
- 2219. [Fixe l'échelle horizontale du texte](#)
- 2220. [Fixe l'élévation du texte](#)
- 2221. [Fixe l'alarme de la structure globale.](#)
- 2222. [Fixe l'animation de la transition entre les pages](#)
- 2223. [Fixe l'apparence d'un champs](#)
- 2224. [Fixe l'echelle horizontale du texte](#)
- 2225. [Fixe l'espacement des caractères.](#) [Fixe l'espacement des caractères](#)
- 2226. [Fixe l'espacement des mots.](#) [Fixe l'espacement des mots](#)
- 2227. [Fixe l'identifiant de group de processus](#)
- 2228. [Fixe l'offset du pointeur de champs](#)
- 2229. [Fixe l'UID effective du processus courant.](#)
- 2230. [Fixe la bufferisation de fichier](#)
- 2231. [Fixe la catégorie de la structure globale.](#)
- 2232. [Fixe la classe de la structure globale.](#)
- 2233. [Fixe la couleur de dessin à un niveau de gris](#)
- 2234. [Fixe la couleur globale d'addition \(? : the global add color\).](#)
- 2235. [Fixe la couleur globale de multiplication \(? : the global multiply color\).](#)
- 2236. [Fixe la couleur pour le style courant de remplissage.](#)
- 2237. [Fixe la description de la structure globale.](#)
- 2238. [Fixe la date de fin de la structure globale.](#)
- 2239. [Fixe la distance entre deux lignes](#)
- 2240. [Fixe la durée de vie de la socket](#)
- 2241. [Fixe la largeur de ligne](#)
- 2242. [Fixe la matrice de texte](#)
- 2243. [Fixe la matrice du texte](#)
- 2244. [Fixe la page courante](#)
- 2245. [Fixe la platitude \(flatness\)](#)
- 2246. [Fixe la position du texte.](#) [Fixe la position du texte](#)
- 2247. [fixe la récurrence annuelle.](#)
- 2248. [Fixe la récurrence hebdomadaire.](#)
- 2249. [fixe la récurrence mensuelle.](#)
- 2250. [Fixe la récurrence quotidienne.](#)
- 2251. [Fixe la récurrence.](#)
- 2252. [Fixe la sévérité minimale du client](#)
- 2253. [Fixe la sévérité minimale du client pour le serveur](#)
- 2254. [Fixe la sévérité minimale du client pour les erreurs](#)
- 2255. [Fixe la sévérité minimale du client pour les messages](#)
- 2256. [Fixe la valeur d'un champs](#)
- 2257. [Fixe la valeur d'une variable d'environnement](#)
- 2258. [Fixe la valeur de la clé /F](#)
- 2259. [Fixe la valeur de la clé /STATUS](#)

- 2260. [Fixe le chemin d'un domaine](#)
- 2261. [Fixe le contexte des actions](#)
- 2262. [Fixe le créateur d'un document PDF](#)
- 2263. [Fixe le domaine par défaut](#)
- 2264. [Fixe le fichier courant, ou la position courante](#)
- 2265. [Fixe le format de date pour les prochaines requêtes.](#)
- 2266. [Fixe le frame courant](#)
- 2267. [Fixe le gestionnaire de référence externes](#)
- 2268. [Fixe le GID effective du processus courant.](#)
- 2269. [Fixe le mode de transition entre les pages](#)
- 2270. [Fixe le motif de pointillé](#)
- 2271. [Fixe le niveau de rapport d'erreurs PHP](#)
- 2272. [Fixe le niveau de sévérité des erreurs.](#)
- 2273. [Fixe le niveau de sévérité des messages d'erreurs.](#)
- 2274. [Fixe le paramètre linecap](#)
- 2275. [Fixe le paramètre linejoin](#)
- 2276. [Fixe le paramètre miter limit](#)
- 2277. [Fixe le point courant](#)
- 2278. [Fixe le point courant relativement](#)
- 2279. [Fixe le style courant de ligne](#)
- 2280. [Fixe le sujet d'un document PDF](#)
- 2281. [Fixe le temps maximum d'exécution d'un script](#)
- 2282. [Fixe le titre d'un document PDF](#)
- 2283. [fixe le titre de la structure globale.](#)
- 2284. [Fixe les dates de début et de fin de la structure globale.](#)
- 2285. [Fixe les limites d'un document PDF](#)
- 2286. [Fixe les mot clés d'un document PDF](#)
- 2287. [Fonctionnement des expressions régulières.](#)
- 2288. [Force le premier caractère d'une chaîne en majuscule.](#)
- 2289. [Force le premier caractère de chaque mot d'une chaîne en majuscule](#)
- 2290. [Formate un nombre par groupe de millier](#)
- 2291. [Formate une date/heure GMT/CUT](#)
- 2292. [Formate une date/heure GMT/CUT en fonction des paramétrages locaux.](#)
- 2293. [Formate une date/heure locale](#)
- 2294. [Formate une date/heure locale avec les options locales.](#)
- 2295. [Fuite de mémoire](#)

G

- 2296. [Générateur de congruence combinée linéaire](#)
- 2297. [Génère un identifiant unique](#)
- 2298. [Génère un message d'erreur utilisateur](#)
- 2299. [Génère un message dans l'historique système.](#)
- 2300. [Génère une meilleure valeur aléatoire.](#)
- 2301. [Génère une valeur aléatoire](#)
- 2302. [Génère une clé](#)
- 2303. [Gestion des colonnes de données binaires](#)
- 2304. [Gestion des colonnes de type LONG](#)
- 2305. [Get result data](#)

H

2306. [Homothétie](#)

I

- 2307. [Identifiant d'objet de l'objet dans l'ancrage](#)
- 2308. [Identifiant d'objet des groupes fils](#)
- 2309. [Identifiant d'objet des parents](#)
- 2310. [Identifiants des ancrages d'un document](#)
- 2311. [Identifie un utilisateur](#)
- 2312. [ids des documents fils d'un groupe](#)
- 2313. [Ignore les actions si le frame n'est pas chargé.](#)
- 2314. [Ignore les mots des moins de N caractères](#)
- 2315. [Importe les variables dans la table des symboles](#)
- 2316. [Importe un objet de grande taille depuis un fichier](#)
- 2317. [Imprime le texte à la ligne suivante](#)
- 2318. [Imprime les crédits de PHP](#)
- 2319. [Imprime un texte à la position courante](#)
- 2320. [Imprime un texte avec des options](#)
- 2321. [Inactive la validation automatique](#)
- 2322. [Inactive le debugger interne de PHP](#)
- 2323. [Inactive le remplissage](#)
- 2324. [Inclidnet une police de caractères](#)
- 2325. [Indique de quoi est capable le navigateur client.](#)
- 2326. [Indique la taille des données à lire dans une colonne de grande taille](#)
- 2327. [Indique les tailles de clé supportées par un algorithme](#)
- 2328. [Indique si le client a abandonné la connexion](#)
- 2329. [Indique si le fichier a été téléchargé par HTTP POST](#)
- 2330. [Indique si le fichier est exécutable](#)
- 2331. [Indique si le fichier est un lien symbolique.](#)
- 2332. [Indique si le fichier est un véritable fichier.](#)
- 2333. [Indique si le nom de fichier est un dossier](#)
- 2334. [Indique si les entêtes HTTP ont déjà été envoyés](#)
- 2335. [Indique si un algorithme fonctionne par bloc](#)
- 2336. [Indique si un mode fonctionne par bloc](#)
- 2337. [Indique si un mode travaille par blocs](#)
- 2338. [Indique si une exception a été émise](#)
- 2339. [Indique si une valeur appartient à un tableau](#)
- 2340. [Indique si une valeur existe](#)
- 2341. [Indique si une variable a été enregistrée dans la session ou pas](#)
- 2342. [Indique un fichier est autorisé en lecture, Indique un fichier est autorisé en lecture](#)
- 2343. [Informations à propos d'une connexion](#)
- 2344. [Initialise la librairie Payflow Pro](#)
- 2345. [Initialise la structure globale d'un flot.](#)
- 2346. [Initialise le générateur de nombres aléatoires](#)
- 2347. [Initialise les données de session](#)
- 2348. [Initialise tous les buffers nécessaires](#)
- 2349. [Initialise toutes les constantes liées au syslog](#)
- 2350. [Initialise un nouveau pointeur vide de LOB/FILE](#)

- 2351. [Initialise une meilleure valeur aléatoire](#)
- 2352. [Initialise une session CURL](#)
- 2353. [Initie une connexion avec une socket](#)
- 2354. [Insère ou modifie une variable de la mémoire partagée.](#)
- 2355. [Insère un document](#)
- 2356. [Insère un document dans un groupe](#)
- 2357. [Insère un groupe](#)
- 2358. [Insère un objet record](#)
- 2359. [Insère une entrée](#)
- 2360. [Insère une valeur](#)
- 2361. [Inverse l'ordre des caractères d'une chaîne.](#)
- 2362. [Inverse modulo](#)

J

- 2363. [Joue l'animation flash à partir du frame courant.](#)
- 2364. [Joue un frame puis stoppe](#)

L

- 2365. [La plus grand valeur aléatoire possible](#)
- 2366. [La plus grande valeur](#)
- 2367. [La plus petite valeur](#)
- 2368. [Le jour de l'année.](#)
- 2369. [Le jour de la semaine.](#)
- 2370. [Lecture du fichier en mode binaire](#)
- 2371. [Lecture du pointeur de fiche courante \(cursorname\)](#)
- 2372. [Libère la mémoire, Libère la mémoire](#)
- 2373. [Libère la mémoire occupée par une police PostScript Type 1.](#)
- 2374. [Libère la mémoire prise par un résultat.](#)
- 2375. [Libère la mémoire réservée par une requête préparée.](#)
- 2376. [Libère le résultat de la mémoire, Libère le résultat de la mémoire](#)
- 2377. [Libère les ressources, Libère les ressources](#)
- 2378. [Libère les ressources associées à un résultat](#)
- 2379. [Libère les ressources prises par un résultat](#)
- 2380. [Libère les ressources utilisées par un résultat](#)
- 2381. [Libère toutes les ressources occupées par un pointeur.](#)
- 2382. [Libère toutes les ressources occupées par une commande.](#)
- 2383. [Libère un résultat](#)
- 2384. [Libère un résultat de la mémoire](#)
- 2385. [Libère un sémaphore](#)
- 2386. [Lie un nom à une socket](#)
- 2387. [Lie une variable PHP à un paramètre Oracle](#)
- 2388. [Linéarise une variable](#)
- 2389. [Lire la taille du bloc de mémoire partagée](#)
- 2390. [Lire le nom du groupe](#)
- 2391. [Lire un paquet WDDX](#)
- 2392. [Lis la liste des boîtes aux lettres, et y recherche une chaîne.](#)
- 2393. [Lis les entêtes EXIF d'une image JPEG](#)
- 2394. [Liste des object ids des objets fils](#)

- 2395. [Liste des object records des objets fils](#)
- 2396. [Liste des utilisateurs actuellement identifiés](#)
- 2397. [Liste l'historique](#)
- 2398. [Liste les bases de données disponibles sur le serveur MySQL.](#)
- 2399. [Liste les bases de données mSQL sur un serveur](#), [Liste les bases de données mSQL sur un serveur](#)
- 2400. [Liste les boîtes aux lettres](#)
- 2401. [Liste les boîtes aux lettres souscrites](#)
- 2402. [Liste les boîtes aux lettres, et retourne le détail pour chacune.](#)
- 2403. [Liste les champs d'une table](#), [Liste les champs d'une table](#)
- 2404. [Liste les champs du résultat MySQL](#)
- 2405. [Liste les champs Informix SQL](#)
- 2406. [Liste les clés étrangères](#)
- 2407. [Liste les colonnes d'une table](#)
- 2408. [Liste les colonnes et leurs droits associés](#)
- 2409. [Liste les colonnes utilisées dans une clé primaire](#)
- 2410. [Liste les fonctions d'une extension](#)
- 2411. [Liste les paramètres des procédures](#)
- 2412. [Liste les procédure stockées](#)
- 2413. [Liste les propriétés des champs SQL](#)
- 2414. [Liste les tables d'une base de données](#)
- 2415. [Liste les tables d'une source.](#)
- 2416. [Liste les tables et leurs privilèges](#)
- 2417. [Liste les tables mSQL sur une base de données](#), [Liste les tables mSQL sur une base de données](#)
- 2418. [Liste les types de données supportés par une source](#)
- 2419. [Liste tous les algorithmes de chiffrement supportés](#)
- 2420. [Liste tous les modes de chiffrement supportés](#)
- 2421. [Liste toutes les boîtes aux lettres souscrites](#)
- 2422. [Liste toutes les classes définies](#)
- 2423. [Lit certains paramètres](#)
- 2424. [Lit certains paramètres numériques](#)
- 2425. [Lit et vérifie un fichier](#)
- 2426. [Lit et/ou modifie le limiteur de cache](#)
- 2427. [Lit l'échelle d'un champs](#)
- 2428. [Lit l'attribut suivant](#), [Lit l'attribut suivant](#)
- 2429. [Lit l'encodage du client](#)
- 2430. [Lit l'entête du message](#)
- 2431. [Lit l'entête d'un message](#)
- 2432. [Lit l'heure locale](#)
- 2433. [Lit l'historique](#)
- 2434. [Lit l'identifiant de repository de la dernière exception](#)
- 2435. [Lit la clé suivante](#), [Lit la clé suivante](#)
- 2436. [Lit la longueur d'un champs](#), [Lit la longueur d'un champs](#)
- 2437. [Lit la première clé](#), [Lit la première clé](#)
- 2438. [Lit la structure d'un message.](#)
- 2439. [Lit la structure de la dernière exception](#)
- 2440. [Lit la taille d'une colonne](#)
- 2441. [Lit la totalité d'un fichier compressé dans un tableau.](#)
- 2442. [Lit la valeur courante d'une option](#)
- 2443. [Lit la valeur courante de l'option quick_print de la librairie UCD.](#)
- 2444. [Lit la valeur d'un champs](#)
- 2445. [Lit la valeur d'une option de configuration](#)

- 2446. [Lit la valeur de la clé /F](#)
- 2447. [Lit la valeur de la clé /STATUS](#)
- 2448. [Lit le contenu d'une colonne](#)
- 2449. [Lit le corps d'un message](#)
- 2450. [Lit le dernier code d'erreur](#)
- 2451. [Lit le dernier message d'erreur](#)
- 2452. [Lit le fichier et renvoie le résultat dans un tableau.](#)
- 2453. [Lit le formatage numérique et monétaire](#)
- 2454. [Lit le message d'erreur de l'analyseur XML](#)
- 2455. [Lit le nom d'un champs](#), [Lit le nom d'un champs](#), [Lit le nom d'un champs](#)
- 2456. [Lit le nom de la base de données courante](#)
- 2457. [Lit le nom de la colonne](#)
- 2458. [Lit le nom du champs suivant](#)
- 2459. [Lit le nom du chiffrement utilisé](#)
- 2460. [Lit le nombre de lignes affectées par une requête immédiate](#)
- 2461. [Lit le numéro de version de Payflow Pro](#)
- 2462. [Lit les données de résultat](#)
- 2463. [Lit les données liées à une clé](#)
- 2464. [Lit les informations sur le champs](#)
- 2465. [Lit les informations à propos de la boîte aux lettres courante.](#)
- 2466. [Lit les informations d'un champs](#)
- 2467. [Lit les informations sur un champs](#)
- 2468. [Lit les informations sur un fichier à partir d'un pointeur de fichier](#)
- 2469. [Lit les informations sur une image](#)
- 2470. [Lit les noms des bases de donné](#)
- 2471. [Lit les options d'un analyseur XML](#)
- 2472. [Lit les paramètres du cookie de session](#)
- 2473. [Lit n bytes d'un objet SLOB](#)
- 2474. [Lit sur une socket](#)
- 2475. [Lit toutes les informations restantes d'un fichier compressé](#)
- 2476. [Lit toutes les lignes d'un tableau, et la met sous la forme d'un tableau HTML.](#)
- 2477. [Lit un bloc](#)
- 2478. [Lit un caractère d'un fichier compressé](#)
- 2479. [Lit un enregistrement dans une base dBase](#)
- 2480. [Lit un enregistrement dans une base, sous la forme d'un tableau associatif.](#)
- 2481. [Lit un fichier compressé en mode binaire](#)
- 2482. [Lit un objet de grande taille](#)
- 2483. [Lit un objet de grande taille en totalité](#)
- 2484. [Lit un résultat](#)
- 2485. [Lit un sommaire des entêtes de messages](#)
- 2486. [Lit une entrée](#)
- 2487. [Lit une entrée du dossier](#)
- 2488. [Lit une image dans un fichier](#)
- 2489. [Lit une ligne](#)
- 2490. [Lit une ligne comme un tableau](#)
- 2491. [Lit une ligne d'un fichier compressé](#)
- 2492. [Lit une ligne d'un fichier compressé et supprime les balises HTML](#)
- 2493. [Lit une ligne dans un objet](#)
- 2494. [Lit une ligne dans un résultat](#), [Lit une ligne dans un résultat](#)
- 2495. [Lit une ligne dans un tableau](#), [Lit une ligne dans un tableau](#), [Lit une ligne dans un tableau](#), [Lit une ligne dans un tableau](#), [Lit une ligne dans un tableau](#)

- 2496. [Lit une ligne dans un tableau associatif](#)
- 2497. [Lit une ligne dans une base Interbase](#)
- 2498. [Lit une ligne dans une base Interbase dans un objet](#)
- 2499. [Lit une ligne de résultat](#)
- 2500. [Lit une ligne de résultat dans un tableau associatif](#)
- 2501. [Lit une ligne de résultat, et la place dans un tableau.](#)
- 2502. [Lit une ligne sous la forme d'un objet](#)
- 2503. [Lit une ligne sous la forme d'un tableau](#)
- 2504. [Lit une valeur](#)
- 2505. [Lit une valeur dans un résultat](#)
- 2506. [Lit une variable dans la mémoire partagée.](#)
- 2507. [Lit/modifie diverses variables internes](#)
- 2508. [logarithme en base 10](#)
- 2509. [Logarithme naturel](#)

M

- 2510. [Mélange les éléments d'un tableau](#)
- 2511. [Marque le fichier pour l'effacement, dans la boîte aux lettres courante.](#)
- 2512. [Message d'erreur](#)
- 2513. [Met tous les caractères en majuscule](#)
- 2514. [Met tous les caractères en minuscule](#)
- 2515. [Mode auto-validation](#)
- 2516. [Modifie certains paramètre numériques](#)
- 2517. [Modifie certains paramètres](#)
- 2518. [Modifie l'échelle](#)
- 2519. [Modifie l'élévation du texte](#)
- 2520. [Modifie l'action javascript d'un champs](#), [Modifie l'action javascript d'un champs](#)
- 2521. [Modifie l'echelle](#)
- 2522. [Modifie l'index d'un champs](#)
- 2523. [Modifie l'origine du système de coordonnées.](#)
- 2524. [Modifie l'origine du système de coordonnées](#)
- 2525. [Modifie la "miter limit"](#)
- 2526. [Modifie la couleur des liens et annotations](#)
- 2527. [Modifie la distance entre les lignes du textes](#)
- 2528. [Modifie la largeur de ligne](#)
- 2529. [Modifie la platitude \(flatness\)](#)
- 2530. [Modifie la prochaine ligne dans le pointeur interne de résultat.](#)
- 2531. [Modifie le bord des liens et annotations](#)
- 2532. [Modifie le contenu d'un objet BLOB](#)
- 2533. [Modifie le contenu d'un objet char](#)
- 2534. [Modifie le fichier d'historique](#)
- 2535. [Modifie le mode par défaut de lecture des valeurs.](#)
- 2536. [Modifie le niveau de gris comme couleur de remplissage.](#)
- 2537. [Modifie le paramètre linejoin](#)
- 2538. [Modifie le point courant](#)
- 2539. [Modifie le pointeur de fichier](#)
- 2540. [Modifie le système de coordonnées](#)
- 2541. [Modifie les attributs d'objet record](#)
- 2542. [Modifie les caractères de remplissage](#)

- 2543. [Modifie les gestionnaires SAX de l'analyseur XSLT](#)
- 2544. [Modifie les paramètres de transaction SESAM](#)
- 2545. [Modifie les paramètres du cookie de session](#)
- 2546. [Modifie les paramètres ODBC.](#)
- 2547. [Modifie les pointillés des liens et annotations](#)
- 2548. [Modifie un événement dans un calendrier MCAL.](#)
- 2549. [Modifie un bit](#)
- 2550. [Modifie un niveau de gris comme couleur de dessin et de remplissage.](#)
- 2551. [Modifie un objet](#)
- 2552. [Modifie une entrée LDAP](#)
- 2553. [Modifie une option d'un champs](#), [Modifie une option d'un champs](#)
- 2554. [Modifie une option de transfert CURL](#)
- 2555. [Modifie une option LDAP](#)
- 2556. [Modifie une propriété d'un composant COM](#), [Modifie une propriété d'un composant COM](#), [Modifie une propriété d'un composant COM](#)
- 2557. [Modifie/remplace le contenu d'un document](#)
- 2558. [Modulo GMP](#)
- 2559. [Morcelle une chaîne](#)
- 2560. [Mot la valeur d'un champs](#)
- 2561. [Multiplication de 2 nombres GMP](#)
- 2562. [Multiplie deux nombres de taille arbitraire](#)

N

- 2563. [Nom de l'utilisateur actuellement identifié](#)
- 2564. [Nom de la base de données](#)
- 2565. [Nombre de colonnes dans un résultat](#)
- 2566. [Nombre de ligne dans un résultat](#)
- 2567. [Nombre GMP aléatoire](#)
- 2568. [Nombre GMP probablement premier](#)
- 2569. [Nomme le frame courant](#)
- 2570. [Numéro de colonne](#)

O

- 2571. [Object id de la racine](#)
- 2572. [Object record de hw document](#)
- 2573. [object records d'un groupe d'enfants](#)
- 2574. [Oouvre un nouveau document PDF](#)
- 2575. [Opposé de nombre GMP](#)
- 2576. [Optimise une base](#)
- 2577. [Options disponibles pour les expressions régulières.](#)
- 2578. [OU exclusif logique](#)
- 2579. [OU logique](#)
- 2580. [Ouverture d'un fichier ou d'une URL](#)
- 2581. [Ouverture d'une base dBase](#)
- 2582. [Ouvre des données scellées](#)
- 2583. [Ouvre la connexion à l'historique système](#)
- 2584. [Ouvre le module de l'algorithme et le mode à utiliser](#)
- 2585. [Ouvre un document FDF](#)

- 2586. [Ouvre un dossier, et récupère un pointeur dessus.](#)
- 2587. [Ouvre un fichier compressé](#)
- 2588. [Ouvre un flot IMAP vers une boîte aux lettres](#)
- 2589. [Ouvre un flot IMAP vers une nouvelle boîte aux lettres.](#)
- 2590. [Ouvre un nouveau document PDF](#)
- 2591. [Ouvre un nouveau fichier Shockwave Flash](#)
- 2592. [Ouvre un objet de grande taille](#)
- 2593. [Ouvre un objet SLOB](#)
- 2594. [Ouvre un pointeur Oracle](#)
- 2595. [Ouvre une base de données](#)
- 2596. [Ouvre une base de données dbm](#)
- 2597. [Ouvre une connexion](#)
- 2598. [Ouvre une connexion à un serveur Informix](#)
- 2599. [Ouvre une connexion à un serveur Ingres](#)
- 2600. [Ouvre une connexion à un serveur MS SQL server](#)
- 2601. [Ouvre une connexion à un serveur MySQL](#)
- 2602. [Ouvre une connexion à un serveur Sybase](#)
- 2603. [Ouvre une connexion à une base de données Interbase.](#)
- 2604. [Ouvre une connexion FTP](#)
- 2605. [Ouvre une connexion Hyperwave](#)
- 2606. [Ouvre une connexion ICAP](#)
- 2607. [Ouvre une connexion MCAL](#)
- 2608. [Ouvre une connexion mSQL](#)
- 2609. [Ouvre une connexion Oracle](#)
- 2610. [Ouvre une connexion persistante à Oracle](#)
- 2611. [Ouvre une connexion persistante à un serveur Informix.](#)
- 2612. [Ouvre une connexion persistante à un serveur Ingres.](#)
- 2613. [Ouvre une connexion persistante à un serveur MS SQL.](#)
- 2614. [Ouvre une connexion persistante à un serveur mSQL](#)
- 2615. [Ouvre une connexion persistante à un serveur MySQL.](#)
- 2616. [Ouvre une connexion persistante à un serveur Sybase](#)
- 2617. [Ouvre une connexion persistante à une base de données](#)
- 2618. [Ouvre une connexion persistante à une base de données Interbase.](#)
- 2619. [Ouvre une connexion persistante à une source de données.](#)
- 2620. [Ouvre une connexion persistante MCAL](#)
- 2621. [Ouvre une connexion SESAM](#)
- 2622. [Ouvre une image créée par les fonctions images PHP.](#)
- 2623. [Ouvre une image GIF](#)
- 2624. [Ouvre une image JPEG, Ouvre une image JPEG](#)
- 2625. [Ouvre une image PNG](#)
- 2626. [Ouvre une image TIFF](#)
- 2627. [Ouvre une socket de connexion Internet ou Unix](#)
- 2628. [Ouvre une socket de connexion Internet ou Unix persistante.](#)

P

- 2629. [PGCD](#)
- 2630. [PGCD étendu](#)
- 2631. [Place le pointeur de résultat à un offset donné](#)
- 2632. [Place un objet sur la scène](#)

- 2633. [Place une image dans la page](#), [Place une image dans la page](#)
- 2634. [Place une image enregistrée dans la page](#)
- 2635. [Plus grande valeur aléatoire possible](#)
- 2636. [Positionne le pointeur de tableau en fin de tableau](#)
- 2637. [Positionne un flag sur un message](#)
- 2638. [Prépare et exécute une requête SQL.](#)
- 2639. [Prépare une chaîne pour une recherche par expression régulière insensible à la casse.](#)
- 2640. [Prépare une clé privée au format PEM](#)
- 2641. [Prépare une commande pour l'exécution](#)
- 2642. [Prépare une expression régulière pour effectuer une recherche insensible à la casse.](#)
- 2643. [Prépare une requête pour lier les paramètres et l'exécuter ultérieurement.](#)
- 2644. [Prépare une requête SQL](#)
- 2645. [Prépare une requête SQL pour l'exécution](#)
- 2646. [Prépare une transaction](#)
- 2647. [Prépare une recherche](#)
- 2648. [Prend une ou plusieurs valeurs, au hasard dans un tableau](#)
- 2649. [Puissance](#), [Puissance](#)
- 2650. [Puissance et modulo](#)

R

- 2651. [Réactive l'ancienne fonction de gestion des erreurs](#)
- 2652. [Récupère une ligne de résultat dans un tableau](#)
- 2653. [Récupère une ligne de résultat dans un tableau énuméré](#)
- 2654. [Récupère tous les entêtes des requêtes HTTP.](#)
- 2655. [Récupère une ligne de résultat dans un objet](#)
- 2656. [Réouvre une connexion MCAL](#)
- 2657. [Répète une chaîne](#)
- 2658. [Réserve un sémaphore](#)
- 2659. [Résolution DNS d'une adresse IP](#)
- 2660. [Racine carrée](#)
- 2661. [Racine carrée avec reste GMP](#)
- 2662. [Racine carrée GMP](#)
- 2663. [Rassemble plusieurs tableaux](#)
- 2664. [Reçoit tous les objets *SNMP* d'un agent](#)
- 2665. [Reçoit un objet *SNMP*](#)
- 2666. [Recherche 0](#), [Recherche 0](#)
- 2667. [Recherche dans tout l'arbre LDAP](#)
- 2668. [Recherche dans un seul niveau](#)
- 2669. [Recherche la dernière occurrence d'un caractère dans une chaîne](#)
- 2670. [Recherche la dernière occurrence d'un caractère dans une chaîne.](#)
- 2671. [Recherche la dernière occurrence d'un caractère dans une chaîne.](#)
- 2672. [Recherche la longueur du premier segment de chaîne qui ne corresponde pas au masque donné.](#)
- 2673. [Recherche la première occurrence d'un caractère.](#)
- 2674. [Recherche par expression régulière insensible à la casse.](#)
- 2675. [Recherche un événement dans le calendrier](#)
- 2676. [Recherche un événement dans le calendrier.](#)
- 2677. [Recherche un message dans le domaine courant](#)
- 2678. [Recherche un objet](#), [Recherche un objet](#)
- 2679. [Recherche un objet dans un groupe](#), [Recherche un objet dans un groupe](#)

- 2680. [Rechercher et remplacer par expression régulière standard.](#)
- 2681. [Recode de fichier à fichier, en fonction de la requête.](#)
- 2682. [Recode une chaîne en fonction de la requête](#)
- 2683. [Recode une fonction grâce à une requête](#)
- 2684. [Reculer d'un frame](#)
- 2685. [Reculer le pointeur courant de tableau](#)
- 2686. [Regroupe tous les éléments d'un tableau dans une chaîne, avec une chaîne de jointure.](#)
- 2687. [Regroupe tous les éléments d'un tableau dans une chaîne, avec une chaîne de jointure.](#)
- 2688. [Rel'che un segment de mémoire partagée](#)
- 2689. [Remet à zéro la session courante](#)
- 2690. [Remet le pointeur interne de tableau au début](#)
- 2691. [Remplace dans une sous partie de chaîne](#)
- 2692. [Remplace le domaine courant](#)
- 2693. [Remplace le domaine lors d'une recherche](#)
- 2694. [Remplace les clés par les valeurs, et les valeurs par les clés](#)
- 2695. [Remplace ou insère une entrée](#)
- 2696. [Remplace toutes les occurrences d'un caractère par un autre.](#)
- 2697. [Remplace toutes les occurrences d'une chaîne par une autre.](#)
- 2698. [Remplace un attribut](#)
- 2699. [Remplace un enregistrement dans une base dBase](#)
- 2700. [Remplace une valeur](#)
- 2701. [Remplacement par expression régulière](#)
- 2702. [Remplacement par expression régulière insensible à la casse.](#)
- 2703. [remplir avec une région avec une couleur spécifique](#)
- 2704. [Remplis et dessine le chemin courant](#)
- 2705. [Remplis le chemin courant, Remplis le chemin courant](#)
- 2706. [Remplis le chemin, dessine le bord et ferme le chemin](#)
- 2707. [Remplis le chemin, et dessine le bord](#)
- 2708. [Remplis les entêtes du document](#)
- 2709. [Remplis, dessine et ferme le chemin courant](#)
- 2710. [Remplit](#)
- 2711. [Renomme un calendrier](#)
- 2712. [Renomme un fichier](#)
- 2713. [Renomme un fichier sur un serveur FTP](#)
- 2714. [Renomme une boîte aux lettres](#)
- 2715. [Renvoie l'espace disque disponible dans le répertoire.](#)
- 2716. [Renvoie l'heure à laquelle l'inode a été accédé pour la dernière fois.](#)
- 2717. [Renvoie la date à laquelle le fichier a été accédé pour la dernière fois.](#)
- 2718. [Renvoie la date de dernière modification du fichier.](#)
- 2719. [Renvoie la ligne courant sur laquelle se trouve le pointeur du fichier et élimine les balises HTML](#)
- 2720. [Renvoie la ligne courante sur laquelle se trouve le pointeur du fichier et cherche dans le résultat les champs CSV](#)
- 2721. [Renvoie la ligne courante sur laquelle se trouve le pointeur du fichier.](#)
- 2722. [Renvoie la position du pointeur du fichier](#)
- 2723. [Renvoie la racine carrée d'un nombre de taille arbitraire.](#)
- 2724. [Renvoie la taille du fichier](#)
- 2725. [Renvoie le caractère que pointe le pointeur du fichier.](#)
- 2726. [Renvoie le nom du dossier](#)
- 2727. [Renvoie le nom du fichier vers lequel pointe un lien symbolique.](#)
- 2728. [Renvoie le nom du possesseur du fichier](#)
- 2729. [Renvoie le numéro d'inode du fichier](#)

- 2730. [Renvoie les informations à propos d'un fichier](#)
- 2731. [Renvoie les informations à propos d'un fichier ou d'un lien symbolique.](#)
- 2732. [Renvoie les informations à propos d'un lien](#)
- 2733. [Renvoie les permissions affectées au fichier.](#)
- 2734. [Remplace le pointeur courant au début du fichier](#)
- 2735. [Remplace le pointeur de fichier au début](#)
- 2736. [Représente un id globale en un id virtuel local](#)
- 2737. [Restaure la valeur de l'option de configuration](#)
- 2738. [Restaure un environnement sauvé](#)
- 2739. [Restaures un environnement](#)
- 2740. [Reste de la division de deux nombres GMP](#)
- 2741. [Retarde l'exécution](#)
- 2742. [Retarde l'exécution en micro-secondes](#)
- 2743. [Retourne à la première entrée du dossier](#)
- 2744. [Retourne chaque paire clé/valeur d'un tableau](#)
- 2745. [Retourne des informations sur les caractères utilisés dans une chaîne.](#)
- 2746. [Retourne des informations sur un groupe.](#) [Retourne des informations sur un groupe](#)
- 2747. [Retourne des informations sur un utilisateur.](#) [Retourne des informations sur un utilisateur](#)
- 2748. [Retourne des informations sur une colonne](#)
- 2749. [Retourne l' UID d'un message.](#)
- 2750. [Retourne l'échelle d'un champ](#)
- 2751. [Retourne l'élément courant d'un tableau](#)
- 2752. [Retourne l'état de la dernière requête SESAM](#)
- 2753. [Retourne l'adresse IP correspondant à un hôte.](#)
- 2754. [Retourne l'ensemble optimal de colonnes, qui permettent de définir uniquement une ligne dans une table](#)
- 2755. [Retourne l'entête d'un message](#)
- 2756. [Retourne l'heure actuelle](#)
- 2757. [Retourne l'ID de l'utilisateur du processus courant.](#)
- 2758. [Retourne l'id du groupe de processus](#)
- 2759. [Retourne l'ID effectif du groupe du processus courant.](#)
- 2760. [Retourne l'identifiant du groupe de processus.](#)
- 2761. [Retourne l'identifiant du premier attribut](#)
- 2762. [Retourne l'identifiant du processus courant](#)
- 2763. [Retourne l'identifiant du processus parent](#)
- 2764. [Retourne l'identifiant généré par la dernière requête INSERT.](#)
- 2765. [retourne l'identifiant maximal de hash](#)
- 2766. [Retourne l'index de l'octet courant d'un analyseur XML](#)
- 2767. [Retourne l'index de la couleur d'un pixel donné](#)
- 2768. [Retourne l'index de la couleur donnée](#)
- 2769. [Retourne l'index de la couleur donnée, ou la plus proche possible.](#)
- 2770. [Retourne l'index de la couleur la plus proche d'une couleur donnée](#)
- 2771. [Retourne l'inode du script](#)
- 2772. [Retourne l'UID du groupe du processus courant.](#)
- 2773. [Retourne l'UID du propriétaire du script actuel](#)
- 2774. [Retourne l'UID effectif de l'utilisateur du processus courant.](#)
- 2775. [Retourne l'URL d'une animation Shockwave Flash](#)
- 2776. [Retourne la classe d'un objet](#)
- 2777. [Retourne la configuration actuelle de l'option magic_quotes_runtime.](#)
- 2778. [Retourne la couleur associée à un index](#)
- 2779. [Retourne la date de dernière modification d'un fichier sur un serveur FTP.](#)

- 2780. [Retourne la date de dernière modification de la page.](#)
- 2781. [Retourne la date/heure](#)
- 2782. [Retourne la dernière erreur \(si elle existe\) qui est survenu lors de la dernière requête.](#)
- 2783. [Retourne la dernière erreur de stmt|conn|global.](#)
- 2784. [Retourne la hauteur d'une image](#)
- 2785. [Retourne la hauteur de l'image](#)
- 2786. [Retourne la hauteur de la police](#)
- 2787. [Retourne la hauteur du A majuscule, et du x minuscule.](#)
- 2788. [Retourne la largeur d'une image, Retourne la largeur d'une image](#)
- 2789. [Retourne la largeur de la police](#)
- 2790. [Retourne la largeur du texte avec la police courante](#)
- 2791. [Retourne la ligne associée](#)
- 2792. [Retourne la ligne suivante dans un tableau](#)
- 2793. [Retourne la liste d'IP correspondants à un hôte.](#)
- 2794. [Retourne la liste des fichiers dans un dossier](#)
- 2795. [Retourne la longueur d'un champs](#)
- 2796. [Retourne la longueur d'une chaîne](#)
- 2797. [Retourne la longueur de la chaîne](#)
- 2798. [Retourne la longueur du champs spécifié.](#)
- 2799. [Retourne la longueur du contenu du buffer de sortie](#)
- 2800. [Retourne la longueur du premier segment qui vérifie le masque.](#)
- 2801. [Retourne la position courante du pointeur interne](#)
- 2802. [Retourne la précision d'un champ](#)
- 2803. [Retourne la prochaine occurrence d'une événement.](#)
- 2804. [Retourne la table de traduction utilisée par htmlspecialchars\(\) et htmlentities\(\).](#)
- 2805. [Retourne la taille d'un champ](#)
- 2806. [Retourne la taille d'un champs](#)
- 2807. [Retourne la taille d'un fichier.](#)
- 2808. [Retourne la taille d'une colonne](#)
- 2809. [Retourne la taille d'une image GIF, JPG ou PNG](#)
- 2810. [Retourne la taille de bloc d'un algorithme, Retourne la taille de bloc d'un algorithme](#)
- 2811. [Retourne la taille de bloc d'un chiffrement](#)
- 2812. [Retourne la taille de bloc du hash](#)
- 2813. [Retourne la taille de chaque colonne d'une ligne de résultat.](#)
- 2814. [Retourne la taille de la chaîne](#)
- 2815. [Retourne la taille de la clé d'un chiffrement](#)
- 2816. [Retourne la taille de la colonne](#)
- 2817. [Retourne la taille du VI d'un algorithme](#)
- 2818. [Retourne la taille du VI utilisé par un couple chiffrement/mode](#)
- 2819. [Retourne la taille imprimée](#)
- 2820. [Retourne la taille interne de stockage d'un champs donné.](#)
- 2821. [Retourne la taille maximale de clé](#)
- 2822. [Retourne la taille maximale de la clé pour un mode](#)
- 2823. [Retourne la valeur ASCII du caractère](#)
- 2824. [Retourne la valeur d'un champs](#)
- 2825. [Retourne la valeur d'un propriété d'un composant COM, Retourne la valeur d'un propriété d'un composant COM](#)
- 2826. [Retourne la valeur d'une colonne dans une ligne lue](#)
- 2827. [Retourne la valeur d'une option de PHP](#)
- 2828. [Retourne la valeur de la variable d'environnement](#)
- 2829. [Retourne la valeur de la variable, au format chaîne](#)

- 2830. [Retourne la valeur de pi](#)
- 2831. [Retourne la valeur numérique \(double\) de la variable.](#)
- 2832. [Retourne la valeur numérique \(integer\) de la variable](#)
- 2833. [Retourne la version courante de CURL](#)
- 2834. [Retourne le chemin canonique absolu](#)
- 2835. [Retourne le chemin du terminal](#)
- 2836. [Retourne le code d'erreur](#)
- 2837. [Retourne le code d'erreur de la dernière requête Informix](#)
- 2838. [Retourne le code d'erreur Oracle](#)
- 2839. [Retourne la configuration actuelle de l'option magic_quotes_gpc.](#)
- 2840. [Retourne le contenu d'un objet BLOB](#)
- 2841. [Retourne le contenu d'un objet char](#)
- 2842. [Retourne le contenu de la variable sqlca.sqlerrd\[0..5\] après une requête.](#)
- 2843. [Retourne le contenu du buffer de sortie](#)
- 2844. [Retourne le couple \(clé ; valeur\) suivant d'une carte donnée.](#)
- 2845. [Retourne le dernier identifiant d'objet](#)
- 2846. [Retourne le dernier message d'erreur du serveur \(min_message_severity?\).](#)
- 2847. [Retourne le dernier message du serveur](#)
- 2848. [Retourne le domaine NIS par défaut](#)
- 2849. [Retourne le dossier de travail](#)
- 2850. [Retourne le fichier courant, ou la position courante](#)
- 2851. [Retourne le flag d'un champs](#)
- 2852. [Retourne le logo](#)
- 2853. [Retourne le logo de Zend](#)
- 2854. [retourne le message d'erreur](#)
- 2855. [Retourne le message d'erreur](#)
- 2856. [Retourne le message d'erreur couran](#)
- 2857. [Retourne le message d'erreur de la dernière requête Informix.](#)
- 2858. [Retourne le message d'erreur Oracle](#)
- 2859. [Retourne le message LDAP de la dernière commande LDAP.](#)
- 2860. [Retourne le niveau d'utilisation des ressources](#)
- 2861. [Retourne le nom d'hôte](#)
- 2862. [Retourne le nom d'hôte correspondant à une IP.](#)
- 2863. [Retourne le nom d'un champ dans le résultat d'une requête](#)
- 2864. [Retourne le nom d'un champs](#), [Retourne le nom d'un champs](#)
- 2865. [Retourne le nom d'une colonne](#), [Retourne le nom d'une colonne](#), [Retourne le nom d'une colonne](#),
[Retourne le nom d'une colonne](#)
- 2866. [Retourne le nom d'une table à partir d'un nom de champs](#)
- 2867. [Retourne le nom d'une table à partir d'un nom de champs.](#)
- 2868. [Retourne le nom de device du terminal](#)
- 2869. [Retourne le nom de l'algorithme](#)
- 2870. [Retourne le nom de la classe d'un objet](#)
- 2871. [Retourne le nom de la colonne de résultat](#)
- 2872. [Retourne le nom de la machine maître pour une carte.](#)
- 2873. [Retourne le nom de la table où se trouve une colonne](#)
- 2874. [Retourne le nom de la table qui contient le champs spécifié](#)
- 2875. [Retourne le nom de login](#)
- 2876. [Retourne le nom de protocole associé au numéro de protocole](#)
- 2877. [Retourne le nom de tty](#)
- 2878. [Retourne le nom du curseur](#)
- 2879. [Retourne le nom du dossier courant](#)

- 2880. [Retourne le nom du hash](#)
- 2881. [Retourne le nom du mode](#)
- 2882. [Retourne le nom du mois.](#)
- 2883. [Retourne le nom du possesseur du script courant.](#)
- 2884. [Retourne le nom du système](#)
- 2885. [Retourne le nombre courant de colonne d'un analyseur XML](#)
- 2886. [Retourne le nombre courant de colonne d'un analyseur XML.](#)
- 2887. [Retourne le nombre d'élément d'un tableau](#)
- 2888. [Retourne le nombre d'arguments passé à la fonction](#)
- 2889. [Retourne le nombre de champs](#)
- 2890. [Retourne le nombre de champs d'un résultat](#)
- 2891. [Retourne le nombre de champs dans un résultat](#), [Retourne le nombre de champs dans un résultat](#),
[Retourne le nombre de champs dans un résultat](#), [Retourne le nombre de champs dans un résultat](#)
- 2892. [Retourne le nombre de champs dans une base filePro](#)
- 2893. [Retourne le nombre de champs dans une base filePro.](#)
- 2894. [Retourne le nombre de champs renvoyés par la dernière requête.](#)
- 2895. [Retourne le nombre de colonne dans un résultat](#)
- 2896. [Retourne le nombre de colonnes](#), [Retourne le nombre de colonnes](#), [Retourne le nombre de colonnes](#)
- 2897. [Retourne le nombre de colonnes dans un résultat](#)
- 2898. [Retourne le nombre de colonnes dans une requête](#)
- 2899. [Retourne le nombre de jour d'un mois.](#)
- 2900. [Retourne le nombre de jour entre le 21 Mars et Pâques. pour une année donnée.](#)
- 2901. [Retourne le nombre de ligne affectées](#)
- 2902. [Retourne le nombre de ligne d'un résultat](#)
- 2903. [Retourne le nombre de lignes](#)
- 2904. [Retourne le nombre de lignes affectées](#)
- 2905. [Retourne le nombre de lignes affectées lors de la dernière requête SQL.](#)
- 2906. [Retourne le nombre de lignes affectées ou retournées par la dernière requête.](#)
- 2907. [Retourne le nombre de lignes affectées par la dernière requête.](#)
- 2908. [Retourne le nombre de lignes affectées par une modification](#)
- 2909. [Retourne le nombre de lignes affectées par une requête.](#)
- 2910. [Retourne le nombre de lignes dans un résultat](#), [Retourne le nombre de lignes dans un résultat](#),
[Retourne le nombre de lignes dans un résultat](#), [Retourne le nombre de lignes dans un résultat](#)
- 2911. [Retourne le nombre de lignes dans un résultat.](#)
- 2912. [Retourne le nombre de message dans la boîte aux lettres courante.](#)
- 2913. [Retourne le nombre de messages récents dans la boîte aux lettres courante.](#)
- 2914. [Retourne le nombre de résultat de la dernière recherche](#)
- 2915. [Retourne le nombre de tuples affectés](#)
- 2916. [Retourne le numéro d'erreur](#)
- 2917. [Retourne le numéro d'erreur LDAP de la dernière commande exécutée.](#)
- 2918. [Retourne le numéro d'ordre d'une carte](#)
- 2919. [Retourne le numéro d'une colonne](#)
- 2920. [Retourne le numéro de colonne](#)
- 2921. [Retourne le numéro de frame courant](#)
- 2922. [Retourne le numéro de la version courante de PHP.](#)
- 2923. [Retourne le numéro de ligne courant d'un analyseur XML.](#)
- 2924. [Retourne le numéro de message d'erreur de la dernière opération MySQL](#)
- 2925. [Retourne le numéro de port](#)
- 2926. [Retourne le numéro de port associé à un service Internet. et un protocole.](#)
- 2927. [Retourne le numéro de processus courant](#)
- 2928. [Retourne le numéro de protocole associé au nom de protocole](#)

- 2929. [Retourne le numéro de séquence de message pour un UID donné.](#)
- 2930. [Retourne le numéro du jour de la semaine.](#)
- 2931. [Retourne le numérod d'erreur courant](#)
- 2932. [Retourne le premier attribut](#)
- 2933. [Retourne le premier couple \(clé ; valeur\) d'une carte donnée.](#)
- 2934. [Retourne le prochain identifiant d'objet libre](#)
- 2935. [Retourne le rectangle entourant un texte et dessiné avec une police PostScript Type1.](#)
- 2936. [retourne le rectangle entourant un texte et dessiné avec une police TrueType](#)
- 2937. [Retourne le reste d'une division entre nombre de taille arbitraire.](#)
- 2938. [Retourne le sémaphore associé à la colonne spécifiée dans le résultat courant.](#)
- 2939. [Retourne le service Internet qui correspond au port et protocole.](#)
- 2940. [Retourne le sid du processus](#)
- 2941. [Retourne le texte associée avec l'erreur générée lors de la dernière requête.](#)
- 2942. [Retourne le timestamp UNIX actuel avec microsecondes.](#)
- 2943. [Retourne le timestamp UNIX actuel.](#)
- 2944. [Retourne le timestamp UNIX d'une date GMT](#)
- 2945. [Retourne le timestamp UNIX d'une date.](#)
- 2946. [Retourne le type d'interface utilisée entre le serveur web et PHP](#)
- 2947. [Retourne le type d'un champ dans le résultat d'une requête](#)
- 2948. [Retourne le type d'un champs](#)
- 2949. [Retourne le type d'un champs donné par index.](#)
- 2950. [Retourne le type de champs](#)
- 2951. [Retourne le type de commande OCI](#)
- 2952. [Retourne le type de données d'une colonne](#)
- 2953. [Retourne le type de fichier](#)
- 2954. [Retourne le type de la colonne de résultat](#)
- 2955. [Retourne le type de la colonne spécifiée dans le résultat courant.](#)
- 2956. [Retourne le type de la variable](#)
- 2957. [Retourne le type numérique d'une colonne](#)
- 2958. [Retourne les ancrages qui pointent sur un objet](#)
- 2959. [Retourne les arguments d'une fonction sous forme de tableau](#)
- 2960. [Retourne les attributs d'une entrée d'un résultat.](#)
- 2961. [Retourne les attributs, et verrouille l'objet](#)
- 2962. [Retourne les bits de status de la connexion](#)
- 2963. [Retourne les données de résultat](#)
- 2964. [Retourne les données enregistrées dans une colonne, à partir d'un résultat, et retourne un objet.](#)
- 2965. [Retourne les enregistrements MX d'un hôte](#)
- 2966. [Retourne les entêtes de tous les messages d'une boîte aux lettres.](#)
- 2967. [Retourne les fils d'un document distant](#)
- 2968. [Retourne les identifiants du groupe du processus courant.](#)
- 2969. [Retourne les informations de statut sur une boîte aux lettres autres que la boîte courante.](#)
- 2970. [Retourne les informations sur le système d'exploitation](#)
- 2971. [Retourne les informations sur une socket](#)
- 2972. [Retourne les lignes résultats sous la forme d'un objet](#)
- 2973. [Retourne les limites système](#)
- 2974. [Retourne les noms des méthodes d'une classe](#)
- 2975. [Retourne les options](#)
- 2976. [Retourne les types d'images supportés par la version courante de PHP](#)
- 2977. [Retourne les valeurs d'un identifiant de résultat](#)
- 2978. [Retourne les valeurs d'un tableau](#)
- 2979. [Retourne les valeurs par défaut des attributs d'une classe.](#)

- 2980. [Retourne plus de détails après une erreur](#)
- 2981. [Retourne tout ou partie d'un résultat SESAM](#)
- 2982. [Retourne toutes les alertes](#)
- 2983. [Retourne toutes les clés d'un tableau](#)
- 2984. [Retourne toutes les entrées](#)
- 2985. [Retourne toutes les entrées d'un résultat](#)
- 2986. [Retourne toutes les erreurs \(si elles existent\).](#)
- 2987. [Retourne toutes les lignes d'un résultat](#)
- 2988. [Retourne toutes les valeurs binaires à partir d'un identifiant de résultat.](#)
- 2989. [Retourne TRUE si la fonction a déjà été définie.](#)
- 2990. [Retourne TRUE si le script a expiré](#)
- 2991. [Retourne un élément de la liste des arguments](#)
- 2992. [Retourne un caractère](#)
- 2993. [Retourne un champs d'un résultat](#)
- 2994. [Retourne un document](#)
- 2995. [Retourne un document distant](#)
- 2996. [Retourne un document texte](#), [Retourne un document texte](#)
- 2997. [Retourne un identifiant de sémaphore](#)
- 2998. [Retourne un identifiant de type de serveur FTP.](#)
- 2999. [Retourne un identifiant positif d'association.](#)
- 3000. [Retourne un message d'erreur](#), [Retourne un message d'erreur](#)
- 3001. [Retourne un ND d'une entrée d'un résultat](#)
- 3002. [Retourne un nouveau pointeur à utiliser pour lier les pointeurs de références](#)
- 3003. [Retourne un objet contenant la structure de date pour le flot courant.](#)
- 3004. [Retourne un résultat](#)
- 3005. [Retourne un tableau avec les noms des fichiers qui sont inclus dans un script](#)
- 3006. [Retourne un tableau avec les noms des fichiers qui sont requis dans un script](#)
- 3007. [Retourne un tableau avec les résultat de la recherche](#)
- 3008. [Retourne un tableau contenant les noms de tous les modules compilés et chargés](#)
- 3009. [Retourne un tableau contenant les tailles de clés acceptées par un algorithme](#)
- 3010. [Retourne un tableau de message après recherche.](#)
- 3011. [Retourne un tableau dont les éléments sont classés en sens inverse.](#)
- 3012. [Retourne un talbeau associatif des propriétés d'un objet](#)
- 3013. [Retourne un timestamp UNIX pour Pâques, à minuit, pour une année donnée.](#)
- 3014. [Retourne une adresse email proprement formatée](#)
- 3015. [Retourne une chaîne contenant les informations de version du serveur.](#)
- 3016. [Retourne une chaîne décrivant une erreur de socket](#)
- 3017. [Retourne une chaîne formatée](#)
- 3018. [Retourne une clé d'un tableau associatif](#)
- 3019. [Retourne une description de l'erreur](#)
- 3020. [Retourne une donnée d'une ligne lue](#)
- 3021. [Retourne une ligne de résultat](#)
- 3022. [Retourne une ligne de résultat sous la forme d'un tableau](#)
- 3023. [Retourne une ligne de résultat sous la forme d'un tableau associatif.](#)
- 3024. [Retourne une ligne sous la forme d'un objet](#), [Retourne une ligne sous la forme d'un objet](#), [Retourne une ligne sous la forme d'un objet](#)
- 3025. [Retourne une ligne sous la forme d'un tableau](#)
- 3026. [Retourne une ligne sous la forme d'un tableau énuméré.](#), [Retourne une ligne sous la forme d'un tableau énuméré.](#)
- 3027. [Retourne une liste d'événement entre deux dates.](#), [Retourne une liste d'événement entre deux dates.](#)
- 3028. [Retourne une liste d'événements qui ont une alarme prévue à une date.](#), [Retourne une liste](#)

[d'événements qui ont une alarme prévue à une date.](#)

3029. [Retourne une partie de la chaîne](#)

3030. [Retourne une section extraite du corps d'un message](#)

3031. [Rotation de la transformation courante](#)

S

3032. [Sélectionne la police et sa taille](#)

3033. [Sélectionne une base de données mSQL](#), [Sélectionne une base de données mSQL](#)

3034. [Sélectionne une base de données MySQL](#)

3035. [Sélectionne une base de données Sybase](#)

3036. [Sélectionne une nouvelle zone pour un dessin ultérieur.](#)

3037. [Sépare le nom du fichier et le nom du dossier.](#)

3038. [Sauve l'environnement courant](#)

3039. [Sauve le dictionnaire personnel dans un fichier](#)

3040. [Sauver un document FDF](#)

3041. [Scelle des données](#)

3042. [Scinde un ND en plusieurs composants](#)

3043. [Scinde une chaîne en morceau, grâce à un délimiteur.](#)

3044. [Scinde une chaîne en plus petits morceaux](#)

3045. [Scinde une chaîne en un tableau, grâce à une expression régulière.](#), [Scinde une chaîne en un tableau, grâce à une expression régulière.](#)

3046. [Se connecte à un serveur LDAP](#)

3047. [Se connecte à un serveur Oracle avec une nouvelle connexion.](#)

3048. [Se lie à un serveur LDAP](#)

3049. [Selectionne la base de données MS SQL](#)

3050. [Selectionne la police courante, et sa taille](#)

3051. [Signe du nombre GMP](#)

3052. [Signe les données](#)

3053. [Sinus](#)

3054. [Souscrit à une boîte aux lettres](#)

3055. [Soustraction de 2 nombres GMP](#)

3056. [Soustrait un nombre de taille arbitraire à un autre.](#)

3057. [Spécifie la syntaxe de lecture des lignes](#)

3058. [Spécifie le nombre maximal de résultat à lire](#)

3059. [strstr\(\), insensible à la casse.](#)

3060. [Suggèregrave:re une orthographe](#)

3061. [Suggère l'orthographe d'un mot](#)

3062. [Supprime la récurrence de la structure globale.](#)

3063. [Supprime tous les événements marqués pour l'effacement](#)

3064. [Supprime un descripteur de LOB](#)

3065. [Supprime un flag sur un message](#)

3066. [Supprime un objet BLOB](#)

3067. [Supprime un objet char](#)

3068. [Supprime un objet SLOB](#)

3069. [Supprime un segment de mémoire partagée sous Unix.](#)

3070. [Supprime une variable dans la session courante](#)

3071. [Symbole de Jacobi](#)

3072. [Symbole de Legendre](#)

3073. [Synchronise avec le serveur PostgreSQL](#)

- 3074. [Synchronise une base de données](#)
- 3075. [Synonyme de `odbc_exec\(\)`](#)

T

- 3076. [Télécharge un fichier depuis un serveur FTP et le sauve dans un fichier déjà ouvert.](#)
- 3077. [Télécharge un fichier depuis un serveur FTP.](#)
- 3078. [Taille d'un document](#)
- 3079. [Tangente](#)
- 3080. [Termine l'action courante](#)
- 3081. [Termine la définition de symbole](#)
- 3082. [Termine la définition du bouton courant.](#)
- 3083. [Termine la liaison avec un serveur LDAP](#)
- 3084. [Termine la souscription à une boîte aux lettres.](#)
- 3085. [Termine le script courant](#)
- 3086. [Termine le suivi d'une connexion PostgreSQL](#)
- 3087. [Termine un cryptage](#)
- 3088. [Termine un document](#)
- 3089. [Termine un flot IMAP](#)
- 3090. [Termine une connexion PostgreSQL](#)
- 3091. [Termine une page.](#) [Termine une page](#)
- 3092. [Termine une section de texte](#)
- 3093. [Termine une transformation](#)
- 3094. [Test un module ouvert](#)
- 3095. [Test le cryptage par bloc d'un algorithme](#)
- 3096. [Teste la fin d'un fichier compressé](#)
- 3097. [Teste la fin du fichier](#)
- 3098. [Teste le cryptage par bloc d'un mode](#)
- 3099. [Teste si la valeur d'une colonne est NULL](#)
- 3100. [Teste si le mode retourne les données par bloc](#)
- 3101. [Teste si un champ est annulable](#)
- 3102. [Teste si un champs est à NULL](#)
- 3103. [Teste un mode](#)
- 3104. [Transforme des données XML](#)
- 3105. [Transforme un texte anglais en timestamp](#)
- 3106. [Transforme une chaîne ASCII en EBCDIC](#)
- 3107. [Transforme une chaîne EBCDIC en ASCII](#)
- 3108. [Transforme une liste de variables en tableau](#)
- 3109. [Transforme une variable en tableau](#)
- 3110. [Translate la transformation courante](#)
- 3111. [Tri d'un tableau avec l'algorithme à "ordre naturel"](#)
- 3112. [Tri d'un tableau avec l'algorithme à "ordre naturel" insensible à la casse](#)
- 3113. [Tri multi-dimensionnel](#)
- 3114. [Trie des messages](#)
- 3115. [Trie en ordre inverse](#)
- 3116. [Trie le tableau](#)
- 3117. [Trie les clés d'un tableau en utilisant une fonction de comparaison définie par l'utilisateur](#)
- 3118. [Trie les valeurs d'un tableau en utilisant une fonction de comparaison définie par l'utilisateur](#)
- 3119. [Trie un tableau en ordre](#)
- 3120. [Trie un tableau en ordre inverse](#)

- 3121. [Trie un tableau en sens inverse et suivant les clés](#)
- 3122. [Trie un tableau en utilisant une fonction de comparaison définie par l'utilisateur.](#)
- 3123. [Trie un tableau suivant les clés](#)
- 3124. [Tronque un fichier](#)
- 3125. [Trouve la première occurrence d'une chaîne](#)
- 3126. [Type de données d'un champs](#)

U

- 3127. [Utilisation des ressources](#)
- 3128. [Utilise les énumérations CORBA](#)
- 3129. [Utilise un analyseur XML à l'intérieur d'un objet.](#)
- 3130. [Utilise une structure CORBA](#)
- 3131. [Utilise une variable PHP pour la phase de définition, dans un SELECT](#)
- 3132. [Utilise une variable PHP pour la phase de définition, dans une commande SELECT.](#)

V

- 3133. [Vérifie le courrier de la boîte aux lettres courante.](#)
- 3134. [Vérifie q'une classe a été définie](#)
- 3135. [Vérifie qu'un identifiant d'objet est dans un groupe.](#)
- 3136. [Vérifie qu'une clé existe](#)
- 3137. [Vérifie qu'une constante existe.](#)
- 3138. [Vérifie que l'année est bissextile.](#)
- 3139. [Vérifie que la méthode existe pour une classe.](#)
- 3140. [Vérifie que le flot IMAP est toujours actif](#)
- 3141. [Vérifie si un fichier existe](#)
- 3142. [Vérifie si une assertion est fausse](#)
- 3143. [Vérifie un mot, Vérifie un mot](#)
- 3144. [Vérifie un mot sans en changer la casse et sans essayer de supprimer les espaces aux extrémités.](#)
- 3145. [Vérifie une signature](#)
- 3146. [Valeur absolue](#)
- 3147. [Valeur absolue GMP](#)
- 3148. [Valide la transaction SESAM en cours](#)
- 3149. [Valide les transactions en cours](#)
- 3150. [Valide une date.](#)
- 3151. [Valide une date/heure](#)
- 3152. [Valide une heure.](#)
- 3153. [Valide une transaction, Valide une transaction, Valide une transaction](#)
- 3154. [Valide une transaction ODBC](#)
- 3155. [Valide une transaction Oracle](#)
- 3156. [Verrouille le fichier](#)
- 3157. [Vide les buffers de sorties](#)

Index des exemples

[\[Notes en ligne\]](#)

A

- 3158. [Accéder aux données du formulaire](#)
- 3159. [Affectation d'une colonne de type "champs multiple"](#)
- 3160. [Affichage de \\$http_post_vars avec each\(\)](#)
- 3161. [Affichage de la liste des attributs d'une entrée](#)
- 3162. [Affichage de texte](#), [Affichage de texte](#)
- 3163. [Affichage multiple d'une image](#)
- 3164. [Affichage sous la forme d'une table HTML](#)
- 3165. [Affiche une liste d'unités organisationnelle](#)
- 3166. [Afficher les fichiers requis et inclus](#)
- 3167. [Afficher toutes les lignes de la table "ordres" sous la forme html](#)
- 3168. [Afficher une erreur SESAM](#)
- 3169. [Afficher une structure XML](#)
- 3170. [Aide OCI](#)
- 3171. [Ajout d'entrées dans un tableau](#)
- 3172. [Ajouter un nouvel attribut](#)
- 3173. [Ajouter une mise en relief](#)
- 3174. [Ajouter une nouvelle ressource](#)
- 3175. [Annulation d'une transaction SESAM](#)
- 3176. [Argument de fonction de lecture](#)
- 3177. [aspell_check](#)
- 3178. [aspell_check_raw](#)
- 3179. [aspell_new](#)
- 3180. [aspell_suggest](#)
- 3181. [Authentification HTTP avec nom d'utilisateur/mot de passe forcé](#)
- 3182. [avoid_error_require_once.php](#)

B

- 3183. [base_convert\(\)](#)

C

- 3184. [Calcul du hash et enregistrement d'un utilisateur](#)
- 3185. [Calcule un hash de type SHA1 et l'affiche au format hexadécimal](#)
- 3186. [cause_error_require.php](#)
- 3187. [Chargement multiple de fichier](#)
- 3188. [Classer un tableau multidimensionnel](#)
- 3189. [classes.inc](#)
- 3190. [Code PHP pour accéder à MaStructure](#)
- 3191. [Code PHP pour accéder à MonEnumeration](#)
- 3192. [Code PHP pour accéder à MonInterface](#)
- 3193. [Code PHP pour gérer les exceptions CORBA](#)

- 3194. [Colorisation d'URL](#)
- 3195. [Compactage d'une chaîne](#)
- 3196. [Compter le nombre de hit d'un utilisateur.](#)
- 3197. [Connaître le titre d'une page distante](#)
- 3198. [Connection à un serveur Informix](#)
- 3199. [Connection au serveur Ovrimos SQL Server et selectionn d'une table système](#)
- 3200. [Connexion à un serveur Ovrimos SQL Server et préparation d'une requête](#)
- 3201. [Connexion à une base SESAM](#)
- 3202. [Conversion d'un nombre GMP en chaîne](#)
- 3203. [Conversion HTML en texte](#)
- 3204. [Création d'images GIF avec PHP](#)
- 3205. [Création d'un document PDF avec pdflib](#)
- 3206. [Création d'un fichier Flash simple, basé sur une entrée de l'utilisateur, et sauvegarde dans une base.](#)
- 3207. [Création d'un nombre GMP](#)
- 3208. [Création d'une base dBase](#)
- 3209. [Création d'une colonne de champs multiples](#)
- 3210. [Créer un nouveau bloc](#)
- 3211. [Creation d'une fonction anonmye avec create_function\(\)](#)
- 3212. [Crypte une valeur avec un TripleDES, en mode ECB.](#)

D

- 3213. [Déconnexion d'une base SESAM](#)
- 3214. [Définition d'une constante](#)
- 3215. [Définition de constantes](#)
- 3216. [dbm](#)
- 3217. [Dernier jour du mois](#)
- 3218. [Division de nombres GMP](#)

E

- 3219. [Eclatement d'une chaîne de recherche.](#)
- 3220. [Ecrire un bloc de mémoire partagée](#)
- 3221. [Effacement d'un bloc de mémoire partagée](#)
- 3222. [Effacer une ressource existante](#)
- 3223. [Encryption d'une valeur avec TripleDES sous 2.4.x en mode ECB](#)
- 3224. [Enregistrer une valeur simple](#)
- 3225. [Enregistrer une variable lorsque @ref{ini.register-globals} est activée](#)
- 3226. [Enregistrer une variable lorsque l'option @ref{ini.track-vars} est activée](#)
- 3227. [Entité externe](#)
- 3228. [Enumérer tous les messages d'erreur LDAP](#)
- 3229. [Envoi de eMail avec des entêtes supplémentaires.](#)
- 3230. [Envoi de mail complexe.](#)
- 3231. [Envoi de mail.](#)
- 3232. [Evaluer un document FDF](#)
- 3233. [exemple de la fonction phpversion\(\)](#)
- 3234. [Exemple](#)
- 3235. [Exemple easter_date\(\)](#)
- 3236. [Exemple ereg\(\)](#)
- 3237. [Exemple fscanf\(\)](#)

- 3238. [Exemple gzopen\(\)](#)
- 3239. [Exemple ibase_connect\(\)](#)
- 3240. [Exemple imap_delete\(\)](#)
- 3241. [Exemple imap_mail_compose\(\)](#)
- 3242. [Exemple imap_mime_header_decode\(\)](#)
- 3243. [Exemple imap_status\(\)](#)
- 3244. [Exemple ip2long\(\)](#)
- 3245. [Exemple mhash_get_hash_name\(\)](#)
- 3246. [Exemple mktime\(\)](#)
- 3247. [Exemple mysql_tablename\(\)](#)
- 3248. [Exemple mysql_data_seek\(\)](#)
- 3249. [Exemple mysql_db_name\(\)](#)
- 3250. [Exemple mysql_num_rows\(\)](#) par crubel@trilizio.org
- 3251. [Exemple mysql_tablename\(\)](#)
- 3252. [Exemple ora_fetch_into\(\)](#)
- 3253. [Exemple pfpro_process_raw\(\)](#)
- 3254. [Exemple php_sapi_name\(\)](#)
- 3255. [Exemple php_uname\(\)](#)
- 3256. [Exemple pspell_add_to_personal\(\)](#)
- 3257. [Exemple realpath\(\)](#)
- 3258. [Exemple socket_set_timeout\(\)](#)
- 3259. [Exemple strftime\(\)](#)
- 3260. [Exemple substr_count\(\)](#)
- 3261. [Exemple wordwrap\(\)](#)
- 3262. [Exemple avec addcslashes\(\)](#)
- 3263. [Exemple avec array\(\)](#)
- 3264. [Exemple avec array_count_values\(\)](#)
- 3265. [Exemple avec array_diff\(\)](#)
- 3266. [Exemple avec array_flip\(\)](#)
- 3267. [Exemple avec array_intersect\(\)](#)
- 3268. [Exemple avec array_keys\(\)](#)
- 3269. [Exemple avec array_merge\(\)](#)
- 3270. [Exemple avec array_merge_recursive\(\)](#)
- 3271. [Exemple avec array_pad\(\)](#)
- 3272. [Exemple avec array_pop\(\)](#)
- 3273. [Exemple avec array_push\(\)](#)
- 3274. [Exemple avec array_rand\(\)](#)
- 3275. [Exemple avec array_reverse\(\)](#)
- 3276. [Exemple avec array_shift\(\)](#)
- 3277. [Exemple avec array_slice\(\)](#)
- 3278. [Exemple avec array_unique\(\)](#)
- 3279. [Exemple avec array_walk\(\)](#)
- 3280. [Exemple avec arsort\(\)](#)
- 3281. [Exemple avec asort\(\)](#)
- 3282. [Exemple avec basename\(\)](#)
- 3283. [Exemple avec chop\(\)](#)
- 3284. [Exemple avec chr\(\)](#)
- 3285. [Exemple avec chunk_split\(\)](#)
- 3286. [Exemple avec compact\(\)](#)
- 3287. [Exemple avec copy\(\)](#)
- 3288. [Exemple avec count\(\)](#)

- 3289. [Exemple avec date\(\)](#)
- 3290. [Exemple avec die\(\)](#)
- 3291. [Exemple avec dir\(\)](#)
- 3292. [Exemple avec dirname\(\)](#)
- 3293. [Exemple avec diskfree\(\)](#)
- 3294. [Exemple avec echo\(\)](#)
- 3295. [Exemple avec ereg_replace\(\)](#), [Exemple avec ereg_replace\(\)](#)
- 3296. [Exemple avec eval\(\) – inclusion de texte](#)
- 3297. [Exemple avec explode\(\)](#)
- 3298. [Exemple avec extract\(\)](#)
- 3299. [Exemple avec fgets\(\) une ligne vide dans un fichier csv sera retournée dans le tableau comme une chaîne vide, et ne sera pas traitée comme une erreur.](#)
- 3300. [Exemple avec fsockopen\(\)](#)
- 3301. [Exemple avec get_browser\(\)](#)
- 3302. [Exemple avec get_object_vars\(\)](#)
- 3303. [Exemple avec getlastmod\(\)](#)
- 3304. [Exemple avec gmdate\(\)](#)
- 3305. [Exemple avec gmstrftime\(\)](#)
- 3306. [Exemple avec htmlspecialchars\(\)](#)
- 3307. [Exemple avec ifx_fetch_row\(\)](#)
- 3308. [Exemple avec ifx_fieldproperties\(\)](#)
- 3309. [Exemple avec imap_append\(\)](#)
- 3310. [Exemple avec imap_createmailbox\(\)](#)
- 3311. [Exemple avec imap_fetch_overview\(\)](#)
- 3312. [Exemple avec imap_getmailboxes\(\)](#)
- 3313. [Exemple avec imap_listmailbox\(\)](#)
- 3314. [Exemple avec imap_mailboxmsginfo\(\)](#)
- 3315. [Exemple avec imap_open\(\)](#)
- 3316. [Exemple avec imap_rfc822_parse_adrlist\(\)](#)
- 3317. [Exemple avec imap_rfc822_write_address\(\)](#)
- 3318. [Exemple avec imap_setflag_full\(\)](#)
- 3319. [Exemple avec implode\(\)](#)
- 3320. [Exemple avec in_array\(\)](#)
- 3321. [Exemple avec krsort\(\)](#)
- 3322. [Exemple avec ksort\(\)](#)
- 3323. [Exemple avec list\(\)](#)
- 3324. [Exemple avec localeconv\(\)](#)
- 3325. [Exemple avec mcrypt_encrypt\(\)](#)
- 3326. [Exemple avec mcrypt_list_algorithms\(\)](#)
- 3327. [Exemple avec mcrypt_list_modes\(\)](#)
- 3328. [Exemple avec mcrypt_module_open\(\)](#)
- 3329. [Exemple avec mdecrypt_generic\(\)](#)
- 3330. [Exemple avec natsort\(\)](#)
- 3331. [Exemple avec openssl_open\(\)](#)
- 3332. [Exemple avec openssl_seal\(\)](#)
- 3333. [Exemple avec openssl_sign\(\)](#)
- 3334. [Exemple avec openssl_verify\(\)](#)
- 3335. [Exemple avec ord\(\)](#)
- 3336. [Exemple avec ovrinos_connect\(\)](#)
- 3337. [Exemple avec ovrinos_result_all\(\)](#)
- 3338. [Exemple avec preg_grep\(\)](#)

- 3339. [Exemple avec pspell_add_to_personal\(\)](#), [Exemple avec pspell_add_to_personal\(\)](#)
- 3340. [Exemple avec pspell_config_create\(\)](#)
- 3341. [Exemple avec pspell_config_ignore\(\)](#)
- 3342. [Exemple avec pspell_config_personal\(\)](#)
- 3343. [Exemple avec pspell_config_repl\(\)](#)
- 3344. [Exemple avec pspell_config_runtogether\(\)](#)
- 3345. [Exemple avec pspell_new_personal\(\)](#)
- 3346. [Exemple avec pspell_store_replacement\(\)](#)
- 3347. [Exemple avec rawurlencode\(\)](#), [Exemple avec rawurlencode\(\)](#)
- 3348. [Exemple avec readline\(\)](#)
- 3349. [Exemple avec rsort\(\)](#)
- 3350. [Exemple avec rtrim\(\) exemple](#)
- 3351. [Exemple avec serialize\(\)](#)
- 3352. [Exemple avec sesam_fetch_array\(\)](#)
- 3353. [Exemple avec sesam_fetch_row\(\)](#)
- 3354. [Exemple avec session_name\(\)](#)
- 3355. [Exemple avec session_set_save_handler\(\)](#)
- 3356. [Exemple avec set_file_buffer\(\)](#)
- 3357. [Exemple avec setcookie\(\)](#)
- 3358. [Exemple avec shuffle\(\)](#)
- 3359. [Exemple avec sort\(\)](#)
- 3360. [Exemple avec split\(\)](#), [Exemple avec split\(\)](#)
- 3361. [Exemple avec sprintf\(\): complété avec des zéros](#)
- 3362. [Exemple avec sprintf\(\): format monétaire](#)
- 3363. [Exemple avec sql_regcase\(\)](#)
- 3364. [Exemple avec sscanf\(\)](#)
- 3365. [Exemple avec str_pad\(\)](#)
- 3366. [Exemple avec str_repeat\(\)](#)
- 3367. [Exemple avec str_replace\(\)](#)
- 3368. [Exemple avec strcasecmp\(\)](#)
- 3369. [Exemple avec strerror\(\)](#)
- 3370. [Exemple avec strrchr\(\)](#)
- 3371. [Exemple avec strstr\(\)](#)
- 3372. [Exemple avec strtok\(\)](#)
- 3373. [Exemple avec strtolower\(\)](#)
- 3374. [Exemple avec strtotime\(\)](#)
- 3375. [Exemple avec strtoupper\(\)](#)
- 3376. [Exemple avec strtr\(\)](#)
- 3377. [Exemple avec substr_replace\(\)](#)
- 3378. [Exemple avec swf_addbuttonrecord\(\)](#)
- 3379. [Exemple avec tempnam\(\)](#)
- 3380. [Exemple avec touch\(\)](#)
- 3381. [Exemple avec ucfirst\(\)](#)
- 3382. [Exemple avec ucwords\(\)](#)
- 3383. [Exemple avec uksort\(\)](#)
- 3384. [Exemple avec unpack\(\)](#)
- 3385. [Exemple avec unserialize\(\)](#)
- 3386. [Exemple avec unset\(\)](#)
- 3387. [Exemple avec urldecode\(\)](#)
- 3388. [Exemple avec urlencode\(\)](#)
- 3389. [Exemple avec urlencode\(\) et htmlentities\(\)](#)

- 3390. [Exemple avec usort\(\)](#)
- 3391. [Exemple avec wordwrap\(\)](#)
- 3392. [Exemple avec fopen\(\)](#)
- 3393. [Exemple avec ImageTypes](#)
- 3394. [Exemple avec le domaine par défaut](#)
- 3395. [Exemple avec les balises méta](#)
- 3396. [Exemple avec les sockets : Client TCP/IP](#)
- 3397. [Exemple avec mcrypt_create_iv](#)
- 3398. [Exemple avec mcrypt_get_cipher_name](#)
- 3399. [Exemple avec Soundex](#)
- 3400. [Exemple avec un formulaire simple](#)
- 3401. [Exemple avec yp_first](#)
- 3402. [Exemple complet avec lien authentifié](#)
- 3403. [Exemple d'authentification HTTP](#)
- 3404. [Exemple d'introduction](#)
- 3405. [Exemple d'ordre NIS](#)
- 3406. [Exemple d'utilisation de la fonction getallheaders\(\)](#)
- 3407. [Exemple DBA](#)
- 3408. [Exemple de callback](#)
- 3409. [Exemple de chaîne Here Doc](#)
- 3410. [Exemple de connexion FTP](#)
- 3411. [Exemple de création de base MySQL](#)
- 3412. [Exemple de fonction variable](#)
- 3413. [Exemple de gestion d'erreur lors de la création d'image \(gracieusement offert par vic@zymsys.com \)](#),
[Exemple de gestion d'erreur lors de la création d'image \(gracieusement offert par vic@zymsys.com \)](#)
- 3414. [Exemple de gestion des erreurs durant création d'image \(gracieusement offert par vic@zymsys.com \)](#)
- 3415. [Exemple de gestion des erreurs durant la création d'une image wbmp \(gracieusement proposé par vic@zymsys.com\)](#)
- 3416. [Exemple de gestion des sorties](#)
- 3417. [Exemple de lecture de ligne](#)
- 3418. [Exemple de maître NIS](#)
- 3419. [Exemple de message du débogueur](#)
- 3420. [Exemple de modification d'option ODBC](#)
- 3421. [Exemple de modification de niveau d'erreur](#)
- 3422. [Exemple de programmation Socket : serveur TCP/IP](#)
- 3423. [Exemple de recherche NIS](#)
- 3424. [Exemple de table de traduction](#)
- 3425. [Exemple getrusage](#)
- 3426. [Exemple MySQL connect](#)
- 3427. [Exemple MySQL close](#)
- 3428. [Exemple Payflow Pro](#)
- 3429. [Exemple pdfclock de la distribution pdflib 2.0](#)
- 3430. [Exemple pdfclock issue de la distribution pdflib](#)
- 3431. [Exemple pour ingres_connect\(\)](#)
- 3432. [Exemple pour ingres_connect\(\) utilisant le lien par défaut](#)
- 3433. [Exemple pour ingres_fetch_array\(\)](#)
- 3434. [Exemple pour ingres_fetch_object\(\)](#)
- 3435. [Exemple pour ingres_fetch_row\(\)](#)
- 3436. [Exemple pour ingres_query\(\)](#)
- 3437. [Exemple simple avec recode_file\(\)](#)
- 3438. [Exemple simple avec recode_string\(\)](#)

- 3439. [Exemple simple ClibPDF](#)
- 3440. [Exemple SWF](#)
- 3441. [Exemples](#)
- 3442. [Exemples avec array_splice\(\)](#)
- 3443. [Exemples avec array_unshift\(\)](#)
- 3444. [Exemples avec array_values\(\)](#)
- 3445. [Exemples avec date\(\) et mktime\(\)](#)
- 3446. [Exemples avec each\(\)](#)
- 3447. [Exemples avec easter_date\(\)](#)
- 3448. [Exemples avec error_log\(\)](#)
- 3449. [Exemples avec error_reporting\(\)](#)
- 3450. [Exemples avec session_cache_limiter\(\)](#)
- 3451. [Exemples avec setcookie\(\)](#)
- 3452. [Exemples de masques invalides](#)
- 3453. [Exemples de masques valides](#)
- 3454. [Expressions régulières](#)
- 3455. [Extraction d'un numéro de page d'une chaîne.](#)
- 3456. [Extraction de tous les numéros de téléphone d'un texte.](#)

F

- 3457. [Factorielle avec GMP](#)
- 3458. [Fermer une connexion Informix](#)
- 3459. [Fermeture d'un bloc de mémoire partagée](#)
- 3460. [Fichier d'exemple IDL : MaStructure](#)
- 3461. [Fichier d'exemple IDL : MonEnumeration](#)
- 3462. [Fichier IDL : MonInterface](#)
- 3463. [Fichier IDL exemple : PlusDeFromage](#)
- 3464. [Fixer la valeur d'une variable d'environnement](#)
- 3465. [Fonctions à nombre d'arguments variable](#)
- 3466. [Fonctions calendrier](#)
- 3467. [foolib.inc](#)
- 3468. [foolib.inc \(corrigé\)](#)
- 3469. [Formulaire de chargement de fichier](#)

G

- 3470. [Générer et intercepter une erreur](#)
- 3471. [getimagesize\(\)](#)
- 3472. [getimagesize\(\) avec une url](#)
- 3473. [getimagesize\(\) qui retourne iptc](#)

I

- 3474. [imagettftext\(\)](#)
- 3475. [Implémentation de array_keys\(\) pour les utilisateurs de php 3](#)
- 3476. [Implémentation de array_values\(\) pour les utilisateurs php 3](#)
- 3477. [in_array\(\) avec le paramètre *strict*](#)
- 3478. [include\(\) en php 3 et php 4](#)

- 3479. [Including a PNG image](#)
- 3480. [Inclusion d'un image GIF](#)
- 3481. [Inclusion d'une image mémoire](#)
- 3482. [Index automatique d'un tableau avec array\(\)](#)
- 3483. [Initialisation d'un tableau](#)
- 3484. [Initialiser une sessions CURL et récupération d'une page web.](#)
- 3485. [Insertion à grande vitesse dans une table](#)
- 3486. [Insertion d'une image dans un fichier PDF](#)
- 3487. [Insertion de valeurs dans la table "catalogue"](#)
- 3488. [Instructions d'installation rapide \(Version Module Apache\)](#)
- 3489. [Introduction à la mémoire partagée](#)

L

- 3490. [Le passage du HTML au PHP](#)
- 3491. [Lecture d'un fichier ligne par ligne](#)
- 3492. [Lire la taille d'un bloc de mémoire partagée](#)
- 3493. [Lire les valeurs de sqlca.sqlerrd\[x\]](#)
- 3494. [Lire un bloc de mémoire partagée](#)
- 3495. [Lire un nom de domaine dans une URL](#)
- 3496. [Liste tous les fichiers du dossier courant](#)
- 3497. [Liste tous les fichiers du dossier courant, sauf . et ..](#)
- 3498. [Liste toutes les lignes de table "phone" sous forme de table HTML](#)
- 3499. [Liste toutes les valeurs avec l'attribut "mail"](#)
- 3500. [Lit un exemple](#)

M

- 3501. [Migration depuis 2.0: concaténation de chaînes](#)
- 3502. [Migration depuis 2.0: valeur retournées, ancienne façon](#)
- 3503. [Migration depuis 2.0: valeur retournées, nouvelle façon](#)
- 3504. [Migration: ancienne syntaxe if..endif](#)
- 3505. [Migration: ancienne syntaxe while..endwhile](#)
- 3506. [Migration: Migration: balises start/end](#)
- 3507. [Migration: nouvelle syntaxe if..endif](#)
- 3508. [Migration: nouvelle syntaxe while..endwhile](#)
- 3509. [Migration: Nouvelles balises PHP](#)
- 3510. [Migration: premières nouvelles balises PHP](#)
- 3511. [Mise à l'échelle](#)
- 3512. [Mise en italique d'un mot dans un texte](#)
- 3513. [Modification d'un attribut](#)
- 3514. [Modification de la version du protocole](#)
- 3515. [Modifier l'attribut de Titre \(Title\)](#)
- 3516. [Modifier l'attribut Title](#)
- 3517. [Modifier les contrôles du serveur LDAP](#)
- 3518. [Modifier les paramètres de configuration SESAM](#)
- 3519. [molddb.xml – Petite base de données moléculaire](#)
- 3520. [mysql fetch array](#)
- 3521. [mysql fetch object](#)
- 3522. [mysql fetch assoc\(\)](#)

3523. [mysql_query\(\)](#), [mysql_query\(\)](#)

N

3524. [Nom de champs et type SQL.](#)

3525. [Nombre de lignes affectées](#)

3526. [Nouvelles balises PHP](#)

O

3527. [OCIColumnName](#)

3528. [OCIColumnSize](#)

3529. [OCIColumnType](#)

3530. [OCIDefineByName](#), [OCIDefineByName](#)

3531. [OCIFetchStatement](#)

3532. [OCILogon](#)

3533. [OCINewDescriptor](#)

3534. [OCINLogon](#)

3535. [OCINumCols](#)

3536. [OCIRowCount](#)

3537. [OCIServerVersion](#)

3538. [ovrimos_result_all\(\)](#) avec meta-information

P

3539. [Parcourir la liste des hash](#)

3540. [parsemolddb.php – analyse molddb.xml et crée un tableau d'objet moléculaires](#)

3541. [Passer en revue une base](#)

3542. [Passer en revue une base de données.](#)

3543. [Petit exemple](#)

3544. [pg_cmdtuples](#)

3545. [Postgres fetch object](#)

3546. [Postgres retourne une ligne](#)

3547. [PostgreSQL fetch array](#)

3548. [Pré remplir un formulaire PDF](#)

3549. [Prépare une requête, l'exécute, et affiche le résultat](#)

3550. [Préparer l'entête d'un document PDF](#)

3551. [Protège des caractères spéciaux](#)

3552. [pspell_check\(\)](#)

3553. [pspell_config_mode\(\)](#)

3554. [pspell_new\(\)](#)

3555. [pspell_new_config\(\)](#)

3556. [pspell_suggest\(\)](#)

Q

3557. [Quelques exemples de chaînes](#)

R

- 3558. [read_exif_data](#)
- 3559. [Recherche LDAP](#), [Recherche LDAP](#)
- 3560. [Recherche les couples de balises HTML \(gourmand\)](#)
- 3561. [Rechercher la taille d'une variable dans la table des symboles](#)
- 3562. [Remplacement de plusieurs valeurs](#)

S

- 3563. [Sauver et restaurer un environnement PDF](#)
- 3564. [Sauver/Restaurer](#)
- 3565. [Stocker des données sur un serveur distant](#)
- 3566. [Suppression d'un attribut](#)

T

- 3567. [Tableau d'index commençant à 1](#)
- 3568. [test_script.php](#)
- 3569. [Traitement des erreurs avec set_error_handler\(\) et trigger_error\(\)](#)
- 3570. [Traitement générique par fonction avec create_function\(\)](#)
- 3571. [Transformation avec une feuille de style XSLT \(avec DOM-XML\)](#)
- 3572. [Translation](#)
- 3573. [Trier plusieurs tableaux](#)
- 3574. [Trouve le mot "web"](#)
- 3575. [Types mysql field](#)

U

- 3576. [Une attaque du système de fichier!](#)
- 3577. [Une erreur de vérification de variable conduit à](#)
- 3578. [Une vérification renforcée](#)
- 3579. [UNIX include_path](#)
- 3580. [users.txt](#)
- 3581. [Using pg_connect arguments](#)
- 3582. [Utilisation d'une connexion UDP](#)
- 3583. [Utilisation dangereuse de variables](#)
- 3584. [Utilisation de parse_str\(\)](#)
- 3585. [Utilisation de syslog\(\)](#)
- 3586. [Utilisation de xslt_process\(\) pour transformer trois](#)
- 3587. [Utilisation de CURL et PHP pour récupérer une page](#)
- 3588. [Utilisation de fonctions anonymes comme fonction de callback](#)
- 3589. [Utilisation de l'option /e](#)
- 3590. [Utilisation de paquets incrémentaux](#)
- 3591. [Utilisation de procédures stockées](#)
- 3592. [Utilisation des constantes FILE et LINE](#)
- 3593. [Utilisation des objets de grande taille \(Large Objects\)](#)
- 3594. [Utilisation des options avec sscanf\(\)](#)
- 3595. [Utiliser dbase_numfields\(\)](#)

- 3596. [Utiliser le contrôle d'erreur dans un script](#)
- 3597. [Utiliser un REF CURSOR issue d'une commande SELECT](#)
- 3598. [Utiliser un REF CURSOR issue d'une procédure enregistrée.](#)
- 3599. [Utiliser une ressource existante](#)
- 3600. [utils.inc](#)

V

- 3601. [Vérification gettext\(\)](#)
- 3602. [Vérification de noms de fichier sécurisée](#)
- 3603. [Vérification d'un mot de passe avec LDAP](#)
- 3604. [Vérification de l'existence de *\\$foo* dans la table des symboles](#)
- 3605. [Vérification de la version du protocole](#)
- 3606. [Validation de fichiers téléchargés](#)
- 3607. [Valider une transaction SESAM](#)
- 3608. [Variables complexes de formulaire](#)

W

- 3609. [wddx_serialize_vars\(\)](#)
- 3610. [Windows include path](#)

X

- 3611. [XML Transtypage XML -> HTML](#)
- 3612. [xmltest.xml](#)
- 3613. [xmltest2.xml](#)

Y

- 3614. [YAZ](#)
-

This document was generated on 2 Février 2001 using the [texi2html](#) translator version 1.52 (extended by [davida@detron.se](#), hacked by [dams@nexen.net](#) for French translation). This document was generated on 2 Février 2001 using the [texi2html](#) translator version 1.52 (extended by [davida@detron.se](#), hacked by [dams@nexen.net](#) for French translation).