

```
1  [PHP]
2
3  ;;;;;;;;;;;;;;;;;;
4  ; About php.ini  ;
5  ;;;;;;;;;;;;;;;;;;
6  ; This file controls many aspects of PHP's behavior.  In order for PHP to
7  ; read it, it must be named 'php.ini'.  PHP looks for it in the current
8  ; working directory, in the path designated by the environment variable
9  ; PHPRC, and in the path that was defined in compile time (in that order).
10 ; Under Windows, the compile-time path is the Windows directory.  The
11 ; path in which the php.ini file is looked for can be overridden using
12 ; the -c argument in command line mode.
13 ;
14 ; The syntax of the file is extremely simple.  Whitespace and Lines
15 ; beginning with a semicolon are silently ignored (as you probably guessed).
16 ; Section headers (e.g. [Foo]) are also silently ignored, even though
17 ; they might mean something in the future.
18 ;
19 ; Directives are specified using the following syntax:
20 ; directive = value
21 ; Directive names are *case sensitive* - foo=bar is different from FOO=bar.
22 ;
23 ; The value can be a string, a number, a PHP constant (e.g. E_ALL or M_PI), one
24 ; of the INI constants (On, Off, True, False, Yes, No and None) or an expression
25 ; (e.g. E_ALL & ~E_NOTICE), or a quoted string ("foo").
26 ;
27 ; Expressions in the INI file are limited to bitwise operators and parentheses:
28 ; |      bitwise OR
29 ; &     bitwise AND
30 ; ~     bitwise NOT
31 ; !     boolean NOT
32 ;
33 ; Boolean flags can be turned on using the values 1, On, True or Yes.
34 ; They can be turned off using the values 0, Off, False or No.
35 ;
36 ; An empty string can be denoted by simply not writing anything after the equal
37 ; sign, or by using the None keyword:
38 ;
39 ; foo =          ; sets foo to an empty string
40 ; foo = none     ; sets foo to an empty string
41 ; foo = "none"  ; sets foo to the string 'none'
42 ;
43 ; If you use constants in your value, and these constants belong to a
44 ; dynamically loaded extension (either a PHP extension or a Zend extension),
45 ; you may only use these constants *after* the line that loads the extension.
46 ;
47 ;
48 ;;;;;;;;;;;;;;;;;;
49 ; About this file ;
50 ;;;;;;;;;;;;;;;;;;
51 ; This is the recommended, PHP 5-style version of the php.ini-dist file.  It
52 ; sets some non standard settings, that make PHP more efficient, more secure,
53 ; and encourage cleaner coding.
54 ;
55 ; The price is that with these settings, PHP may be incompatible with some
56 ; applications, and sometimes, more difficult to develop with.  Using this
57 ; file is warmly recommended for production sites.  As all of the changes from
```

```
58 ; the standard settings are thoroughly documented, you can go over each one,
59 ; and decide whether you want to use it or not.
60 ;
61 ; For general information about the php.ini file, please consult the php.ini-dist
62 ; file, included in your PHP distribution.
63 ;
64 ; This file is different from the php.ini-dist file in the fact that it features
65 ; different values for several directives, in order to improve performance, while
66 ; possibly breaking compatibility with the standard out-of-the-box behavior of
67 ; PHP. Please make sure you read what's different, and modify your scripts
68 ; accordingly, if you decide to use this file instead.
69 ;
70 ; - register_long_arrays = Off      [Performance]
71 ;   Disables registration of the older (and deprecated) long predefined array
72 ;   variables ($HTTP_*_VARS). Instead, use the superglobals that were
73 ;   introduced in PHP 4.1.0
74 ; - display_errors = Off           [Security]
75 ;   With this directive set to off, errors that occur during the execution of
76 ;   scripts will no longer be displayed as a part of the script output, and thus,
77 ;   will no longer be exposed to remote users. With some errors, the error message
78 ;   content may expose information about your script, web server, or database
79 ;   server that may be exploitable for hacking. Production sites should have this
80 ;   directive set to off.
81 ; - log_errors = On                [Security]
82 ;   This directive complements the above one. Any errors that occur during the
83 ;   execution of your script will be logged (typically, to your server's error log,
84 ;   but can be configured in several ways). Along with setting display_errors to
85 ;   off,
86 ;   this setup gives you the ability to fully understand what may have gone wrong,
87 ;   without exposing any sensitive information to remote users.
88 ; - output_buffering = 4096       [Performance]
89 ;   Set a 4KB output buffer. Enabling output buffering typically results in less
90 ;   writes, and sometimes less packets sent on the wire, which can often lead to
91 ;   better performance. The gain this directive actually yields greatly depends
92 ;   on which Web server you're working with, and what kind of scripts you're using.
93 ; - register_argc_argv = Off       [Performance]
94 ;   Disables registration of the somewhat redundant $argv and $argc global
95 ;   variables.
96 ; - magic_quotes_gpc = Off        [Performance]
97 ;   Input data is no longer escaped with slashes so that it can be sent into
98 ;   SQL databases without further manipulation. Instead, you should use the
99 ;   database vendor specific escape string function on each input element you
100 ;   wish to send to a database.
101 ; - variables_order = "GPCS"      [Performance]
102 ;   The environment variables are not hashed into the $_ENV. To access
103 ;   environment variables, you can use getenv() instead.
104 ; - error_reporting = E_ALL       [Code Cleanliness, Security(?)]
105 ;   By default, PHP suppresses errors of type E_NOTICE. These error messages
106 ;   are emitted for non-critical errors, but that could be a symptom of a bigger
107 ;   problem. Most notably, this will cause error messages about the use
108 ;   of uninitialized variables to be displayed.
109 ; - allow_call_time_pass_reference = Off [Code cleanliness]
110 ;   It's not possible to decide to force a variable to be passed by reference
111 ;   when calling a function. The PHP 4 style to do this is by making the
112 ;   function require the relevant argument by reference.
113 ; - short_open_tag = Off          [Portability]
114 ;   Using short tags is discouraged when developing code meant for redistribution
```

```
114 ; since short tags may not be supported on the target server.
115
116 ;;;;;;;;;;;;;;;;;;;;;;;;;
117 ; Language Options ;
118 ;;;;;;;;;;;;;;;;;;;;;;;;;
119
120 ; Enable the PHP scripting language engine under Apache.
121 engine = On
122
123 ; Enable compatibility mode with Zend Engine 1 (PHP 4.x)
124 zend.zel_compatibility_mode = Off
125
126 ; Allow the <? tag. Otherwise, only <?php and <script> tags are recognized.
127 ; NOTE: Using short tags should be avoided when developing applications or
128 ; libraries that are meant for redistribution, or deployment on PHP
129 ; servers which are not under your control, because short tags may not
130 ; be supported on the target server. For portable, redistributable code,
131 ; be sure not to use short tags.
132 short_open_tag = Off
133
134 ; Allow ASP-style <% %> tags.
135 asp_tags = Off
136
137 ; The number of significant digits displayed in floating point numbers.
138 precision = 14
139
140 ; Enforce year 2000 compliance (will cause problems with non-compliant browsers)
141 y2k_compliance = On
142
143 ; Output buffering allows you to send header lines (including cookies) even
144 ; after you send body content, at the price of slowing PHP's output layer a
145 ; bit. You can enable output buffering during runtime by calling the output
146 ; buffering functions. You can also enable output buffering for all files by
147 ; setting this directive to On. If you wish to limit the size of the buffer
148 ; to a certain size - you can use a maximum number of bytes instead of 'On', as
149 ; a value for this directive (e.g., output_buffering=4096).
150 output_buffering = 4096
151
152 ; You can redirect all of the output of your scripts to a function. For
153 ; example, if you set output_handler to "mb_output_handler", character
154 ; encoding will be transparently converted to the specified encoding.
155 ; Setting any output handler automatically turns on output buffering.
156 ; Note: People who wrote portable scripts should not depend on this ini
157 ; directive. Instead, explicitly set the output handler using ob_start().
158 ; Using this ini directive may cause problems unless you know what script
159 ; is doing.
160 ; Note: You cannot use both "mb_output_handler" with "ob_iconv_handler"
161 ; and you cannot use both "ob_gzhandler" and "zlib.output_compression".
162 ; Note: output_handler must be empty if this is set 'On' !!!!
163 ; Instead you must use zlib.output_handler.
164 ;output_handler =
165
166 ; Transparent output compression using the zlib library
167 ; Valid values for this option are 'off', 'on', or a specific buffer size
168 ; to be used for compression (default is 4KB)
169 ; Note: Resulting chunk size may vary due to nature of compression. PHP
170 ; outputs chunks that are few hundreds bytes each as a result of
```

```
171 ; compression. If you prefer a larger chunk size for better
172 ; performance, enable output_buffering in addition.
173 ; Note: You need to use zlib.output_handler instead of the standard
174 ; output_handler, or otherwise the output will be corrupted.
175 zlib.output_compression = Off
176 ;zlib.output_compression_level = -1
177
178 ; You cannot specify additional output handlers if zlib.output_compression
179 ; is activated here. This setting does the same as output_handler but in
180 ; a different order.
181 ;zlib.output_handler =
182
183 ; Implicit flush tells PHP to tell the output layer to flush itself
184 ; automatically after every output block. This is equivalent to calling the
185 ; PHP function flush() after each and every call to print() or echo() and each
186 ; and every HTML block. Turning this option on has serious performance
187 ; implications and is generally recommended for debugging purposes only.
188 implicit_flush = Off
189
190 ; The unserialize callback function will be called (with the undefined class'
191 ; name as parameter), if the unserializer finds an undefined class
192 ; which should be instantiated.
193 ; A warning appears if the specified function is not defined, or if the
194 ; function doesn't include/implement the missing class.
195 ; So only set this entry, if you really want to implement such a
196 ; callback-function.
197 unserialize_callback_func=
198
199 ; When floats & doubles are serialized store serialize_precision significant
200 ; digits after the floating point. The default value ensures that when floats
201 ; are decoded with unserialize, the data will remain the same.
202 serialize_precision = 100
203
204 ; Whether to enable the ability to force arguments to be passed by reference
205 ; at function call time. This method is deprecated and is likely to be
206 ; unsupported in future versions of PHP/Zend. The encouraged method of
207 ; specifying which arguments should be passed by reference is in the function
208 ; declaration. You're encouraged to try and turn this option Off and make
209 ; sure your scripts work properly with it in order to ensure they will work
210 ; with future versions of the language (you will receive a warning each time
211 ; you use this feature, and the argument will be passed by value instead of by
212 ; reference).
213 allow_call_time_pass_reference = Off
214
215 ;
216 ; Safe Mode
217 ;
218 safe_mode = Off
219
220 ; By default, Safe Mode does a UID compare check when
221 ; opening files. If you want to relax this to a GID compare,
222 ; then turn on safe_mode_gid.
223 safe_mode_gid = Off
224
225 ; When safe_mode is on, UID/GID checks are bypassed when
226 ; including files from this directory and its subdirectories.
227 ; (directory must also be in include_path or full path must
```

```
228 ; be used when including)
229 safe_mode_include_dir =
230
231 ; When safe_mode is on, only executables located in the safe_mode_exec_dir
232 ; will be allowed to be executed via the exec family of functions.
233 safe_mode_exec_dir =
234
235 ; Setting certain environment variables may be a potential security breach.
236 ; This directive contains a comma-delimited list of prefixes. In Safe Mode,
237 ; the user may only alter environment variables whose names begin with the
238 ; prefixes supplied here. By default, users will only be able to set
239 ; environment variables that begin with PHP_ (e.g. PHP_FOO=BAR).
240 ;
241 ; Note: If this directive is empty, PHP will let the user modify ANY
242 ; environment variable!
243 safe_mode_allowed_env_vars = PHP_
244
245 ; This directive contains a comma-delimited list of environment variables that
246 ; the end user won't be able to change using putenv(). These variables will be
247 ; protected even if safe_mode_allowed_env_vars is set to allow to change them.
248 safe_mode_protected_env_vars = LD_LIBRARY_PATH
249
250 ; open_basedir, if set, limits all file operations to the defined directory
251 ; and below. This directive makes most sense if used in a per-directory
252 ; or per-virtualhost web server configuration file. This directive is
253 ; *NOT* affected by whether Safe Mode is turned On or Off.
254 ;open_basedir =
255
256 ; This directive allows you to disable certain functions for security reasons.
257 ; It receives a comma-delimited list of function names. This directive is
258 ; *NOT* affected by whether Safe Mode is turned On or Off.
259 disable_functions =
260
261 ; This directive allows you to disable certain classes for security reasons.
262 ; It receives a comma-delimited list of class names. This directive is
263 ; *NOT* affected by whether Safe Mode is turned On or Off.
264 disable_classes =
265
266 ; Colors for Syntax Highlighting mode. Anything that's acceptable in
267 ; <span style="color: ???????"> would work.
268 ;highlight.string = #DD0000
269 ;highlight.comment = #FF9900
270 ;highlight.keyword = #007700
271 ;highlight.bg = #FFFFFF
272 ;highlight.default = #0000BB
273 ;highlight.html = #000000
274
275 ; If enabled, the request will be allowed to complete even if the user aborts
276 ; the request. Consider enabling it if executing long request, which may end up
277 ; being interrupted by the user or a browser timing out.
278 ; ignore_user_abort = On
279
280 ; Determines the size of the realpath cache to be used by PHP. This value should
281 ; be increased on systems where PHP opens many files to reflect the quantity of
282 ; the file operations performed.
283 ; realpath_cache_size=16k
284
```

```
285 ; Duration of time, in seconds for which to cache realpath information for a given
286 ; file or directory. For systems with rarely changing files, consider increasing this
287 ; value.
288 ; realpath_cache_ttl=120
289
290 ;
291 ; Misc
292 ;
293 ; Decides whether PHP may expose the fact that it is installed on the server
294 ; (e.g. by adding its signature to the Web server header). It is no security
295 ; threat in any way, but it makes it possible to determine whether you use PHP
296 ; on your server or not.
297 expose_php = On
298
299
300 ;;;;;;;;;;;;;;
301 ; Resource Limits ;
302 ;;;;;;;;;;;;;;
303
304 max_execution_time = 30 ; Maximum execution time of each script, in seconds
305 max_input_time = 60 ; Maximum amount of time each script may spend parsing request
    data
306 ;max_input_nesting_level = 64 ; Maximum input variable nesting level
307 memory_limit = 128M ; Maximum amount of memory a script may consume (128MB)
308
309
310 ;;;;;;;;;;;;;;
311 ; Error handling and logging ;
312 ;;;;;;;;;;;;;;
313
314 ; error_reporting is a bit-field. Or each number up to get desired error
315 ; reporting level
316 ; E_ALL - All errors and warnings (doesn't include E_STRICT)
317 ; E_ERROR - fatal run-time errors
318 ; E_RECOVERABLE_ERROR - almost fatal run-time errors
319 ; E_WARNING - run-time warnings (non-fatal errors)
320 ; E_PARSE - compile-time parse errors
321 ; E_NOTICE - run-time notices (these are warnings which often result
322 ; from a bug in your code, but it's possible that it was
323 ; intentional (e.g., using an uninitialized variable and
324 ; relying on the fact it's automatically initialized to an
325 ; empty string)
326 ; E_STRICT - run-time notices, enable to have PHP suggest changes
327 ; to your code which will ensure the best interoperability
328 ; and forward compatibility of your code
329 ; E_CORE_ERROR - fatal errors that occur during PHP's initial startup
330 ; E_CORE_WARNING - warnings (non-fatal errors) that occur during PHP's
331 ; initial startup
332 ; E_COMPILE_ERROR - fatal compile-time errors
333 ; E_COMPILE_WARNING - compile-time warnings (non-fatal errors)
334 ; E_USER_ERROR - user-generated error message
335 ; E_USER_WARNING - user-generated warning message
336 ; E_USER_NOTICE - user-generated notice message
337 ;
338 ; Examples:
339 ;
340 ; - Show all errors, except for notices and coding standards warnings
```

```
341 ;
342 error_reporting = E_ALL & ~E_NOTICE
343 ;
344 ; - Show all errors, except for notices
345 ;
346 ;error_reporting = E_ALL & ~E_NOTICE | E_STRICT
347 ;
348 ; - Show only errors
349 ;
350 ;error_reporting = E_COMPILE_ERROR|E_RECOVERABLE_ERROR|E_ERROR|E_CORE_ERROR
351 ;
352 ; - Show all errors, except coding standards warnings
353 ;
354 ;error_reporting = E_ALL
355
356 ; Print out errors (as a part of the output). For production web sites,
357 ; you're strongly encouraged to turn this feature off, and use error logging
358 ; instead (see below). Keeping display_errors enabled on a production web site
359 ; may reveal security information to end users, such as file paths on your Web
360 ; server, your database schema or other information.
361 ;
362 ; possible values for display_errors:
363 ;
364 ; Off - Do not display any errors
365 ; stderr - Display errors to STDERR (affects only CGI/CLI binaries!)
366 ; On or stdout - Display errors to STDOUT (default)
367 ;
368 ; To output errors to STDERR with CGI/CLI:
369 ;display_errors = "stderr"
370 ;
371 ; Default
372 ;
373 display_errors = Off
374
375 ; Even when display_errors is on, errors that occur during PHP's startup
376 ; sequence are not displayed. It's strongly recommended to keep
377 ; display_startup_errors off, except for when debugging.
378 display_startup_errors = Off
379
380 ; Log errors into a log file (server-specific log, stderr, or error_log (below))
381 ; As stated above, you're strongly advised to use error logging in place of
382 ; error displaying on production web sites.
383 log_errors = On
384
385 ; Set maximum length of log_errors. In error_log information about the source is
386 ; added. The default is 1024 and 0 allows to not apply any maximum length at all.
387 log_errors_max_len = 1024
388
389 ; Do not log repeated messages. Repeated errors must occur in same file on same
390 ; line unless ignore_repeated_source is set true.
391 ignore_repeated_errors = Off
392
393 ; Ignore source of message when ignoring repeated messages. When this setting
394 ; is On you will not log errors with repeated messages from different files or
395 ; source lines.
396 ignore_repeated_source = Off
397
```



```
398 ; If this parameter is set to Off, then memory leaks will not be shown (on
399 ; stdout or in the log). This has only effect in a debug compile, and if
400 ; error reporting includes E_WARNING in the allowed list
401 report_memleaks = On
402
403 ;report zend_debug = 0
404
405 ; Store the last error/warning message in $php_errormsg (boolean).
406 track_errors = Off
407
408 ; Turn off normal error reporting and emit XML-RPC error XML
409 ;xmlrpc_errors = 0
410 ; An XML-RPC faultCode
411 ;xmlrpc_error_number = 0
412
413 ; Disable the inclusion of HTML tags in error messages.
414 ; Note: Never use this feature for production boxes.
415 ;html_errors = Off
416
417 ; If html_errors is set On PHP produces clickable error messages that direct
418 ; to a page describing the error or function causing the error in detail.
419 ; You can download a copy of the PHP manual from http://www.php.net/docs.php
420 ; and change docref_root to the base URL of your local copy including the
421 ; leading '/'. You must also specify the file extension being used including
422 ; the dot.
423 ; Note: Never use this feature for production boxes.
424 ;docref_root = "/phpmanual/"
425 ;docref_ext = .html
426
427 ; String to output before an error message.
428 ;error_prepend_string = "<font color=#ff0000>"
429
430 ; String to output after an error message.
431 ;error_append_string = "</font>"
432
433 ; Log errors to specified file.
434 ;error_log = filename
435
436 ; Log errors to syslog (Event Log on NT, not valid in Windows 95).
437 error_log = syslog
438
439
440 ;;;;;;;;;;;;;;
441 ; Data Handling ;
442 ;;;;;;;;;;;;;;
443 ;
444 ; Note - track_vars is ALWAYS enabled as of PHP 4.0.3
445
446 ; The separator used in PHP generated URLs to separate arguments.
447 ; Default is "&".
448 ;arg_separator.output = "&"
449
450 ; List of separator(s) used by PHP to parse input URLs into variables.
451 ; Default is "&".
452 ; NOTE: Every character in this directive is considered as separator!
453 ;arg_separator.input = ";"
454
```



```
455 ; This directive describes the order in which PHP registers GET, POST, Cookie,
456 ; Environment and Built-in variables (G, P, C, E & S respectively, often
457 ; referred to as EGPCS or GPC). Registration is done from left to right, newer
458 ; values override older values.
459 variables_order = "GPCS"
460
461 ; Whether or not to register the EGPCS variables as global variables. You may
462 ; want to turn this off if you don't want to clutter your scripts' global scope
463 ; with user data. This makes most sense when coupled with track_vars - in which
464 ; case you can access all of the GPC variables through the $HTTP_*_VARS[],
465 ; variables.
466 ;
467 ; You should do your best to write your scripts so that they do not require
468 ; register_globals to be on; Using form variables as globals can easily lead
469 ; to possible security problems, if the code is not very well thought of.
470 register_globals = Off
471
472 ; Whether or not to register the old-style input arrays, HTTP_GET_VARS
473 ; and friends. If you're not using them, it's recommended to turn them off,
474 ; for performance reasons.
475 register_long_arrays = Off
476
477 ; This directive tells PHP whether to declare the argv&argc variables (that
478 ; would contain the GET information). If you don't use these variables, you
479 ; should turn it off for increased performance.
480 register_argc_argv = Off
481
482 ; When enabled, the SERVER and ENV variables are created when they're first
483 ; used (Just In Time) instead of when the script starts. If these variables
484 ; are not used within a script, having this directive on will result in a
485 ; performance gain. The PHP directives register_globals, register_long_arrays,
486 ; and register_argc_argv must be disabled for this directive to have any affect.
487 auto_globals_jit = On
488
489 ; Maximum size of POST data that PHP will accept.
490 post_max_size = 8M
491
492 ; Magic quotes
493 ;
494
495 ; Magic quotes for incoming GET/POST/Cookie data.
496 magic_quotes_gpc = Off
497
498 ; Magic quotes for runtime-generated data, e.g. data from SQL, from exec(), etc.
499 magic_quotes_runtime = Off
500
501 ; Use Sybase-style magic quotes (escape ' with '' instead of \').
502 magic_quotes_sybase = Off
503
504 ; Automatically add files before or after any PHP document.
505 auto_prepend_file =
506 auto_append_file =
507
508 ; As of 4.0b4, PHP always outputs a character encoding by default in
509 ; the Content-type: header. To disable sending of the charset, simply
510 ; set it to be empty.
511 ;
```

```
512 ; PHP's built-in default is text/html
513 default_mimetype = "text/html"
514 ;default_charset = "iso-8859-1"
515
516 ; Always populate the $HTTP_RAW_POST_DATA variable.
517 ;always_populate_raw_post_data = On
518
519
520 ;;;;;;;;;;;;;;;;;;;;;;;;;;
521 ; Paths and Directories ;
522 ;;;;;;;;;;;;;;;;;;;;;;;;;;
523
524 ; UNIX: "/path1:/path2"
525 ;include_path = "./php/includes"
526 ;
527 ; Windows: "\path1;\path2"
528 include_path = ".;c:\php\includes"
529
530 ; The root of the PHP pages, used only if nonempty.
531 ; if PHP was not compiled with FORCE_REDIRECT, you SHOULD set doc_root
532 ; if you are running php as a CGI under any web server (other than IIS)
533 ; see documentation for security issues. The alternate is to use the
534 ; cgi.force_redirect configuration below
535 doc_root =
536
537 ; The directory under which PHP opens the script using ~/username used only
538 ; if nonempty.
539 user_dir =
540
541 ; Directory in which the loadable extensions (modules) reside.
542 extension_dir = "./"
543
544 ; Whether or not to enable the dl() function. The dl() function does NOT work
545 ; properly in multithreaded servers, such as IIS or Zeus, and is automatically
546 ; disabled on them.
547 enable_dl = On
548
549 ; cgi.force_redirect is necessary to provide security running PHP as a CGI under
550 ; most web servers. Left undefined, PHP turns this on by default. You can
551 ; turn it off here AT YOUR OWN RISK
552 ; **You CAN safely turn this off for IIS, in fact, you MUST.**
553 ; cgi.force_redirect = 1
554
555 ; if cgi.nph is enabled it will force cgi to always sent Status: 200 with
556 ; every request.
557 ; cgi.nph = 1
558
559 ; if cgi.force_redirect is turned on, and you are not running under Apache or Netscape
560 ; (iPlanet) web servers, you MAY need to set an environment variable name that PHP
561 ; will look for to know it is OK to continue execution. Setting this variable MAY
562 ; cause security issues, KNOW WHAT YOU ARE DOING FIRST.
563 ; cgi.redirect_status_env = ;
564
565 ; cgi.fix_pathinfo provides *real* PATH_INFO/PATH_TRANSLATED support for CGI. PHP's
566 ; previous behaviour was to set PATH_TRANSLATED to SCRIPT_FILENAME, and to not grok
567 ; what PATH_INFO is. For more information on PATH_INFO, see the cgi specs. Setting
568 ; this to 1 will cause PHP CGI to fix it's paths to conform to the spec. A setting
```

```
569 ; of zero causes PHP to behave as before. Default is 1. You should fix your scripts
570 ; to use SCRIPT_FILENAME rather than PATH_TRANSLATED.
571 ; cgi.fix_pathinfo=1
572
573 ; FastCGI under IIS (on WINNT based OS) supports the ability to impersonate
574 ; security tokens of the calling client. This allows IIS to define the
575 ; security context that the request runs under. mod_fastcgi under Apache
576 ; does not currently support this feature (03/17/2002)
577 ; Set to 1 if running under IIS. Default is zero.
578 ; fastcgi.impersonate = 1;
579
580 ; Disable logging through FastCGI connection
581 ; fastcgi.logging = 0
582
583 ; cgi.rfc2616_headers configuration option tells PHP what type of headers to
584 ; use when sending HTTP response code. If it's set 0 PHP sends Status: header that
585 ; is supported by Apache. When this option is set to 1 PHP will send
586 ; RFC2616 compliant header.
587 ; Default is zero.
588 ;cgi.rfc2616_headers = 0
589
590
591 ;;;;;;;;;;;;;;
592 ; File Uploads ;
593 ;;;;;;;;;;;;;;
594
595 ; Whether to allow HTTP file uploads.
596 file_uploads = On
597
598 ; Temporary directory for HTTP uploaded files (will use system default if not
599 ; specified).
600 ;upload_tmp_dir =
601
602 ; Maximum allowed size for uploaded files.
603 upload_max_filesize = 2M
604
605
606 ;;;;;;;;;;;;;;
607 ; Fopen wrappers ;
608 ;;;;;;;;;;;;;;
609
610 ; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
611 allow_url_fopen = On
612
613 ; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
614 allow_url_include = Off
615
616 ; Define the anonymous ftp password (your email address)
617 ;from="john@doe.com"
618
619 ; Define the User-Agent string
620 ; user_agent="PHP"
621
622 ; Default timeout for socket based streams (seconds)
623 default_socket_timeout = 60
624
625 ; If your scripts have to deal with files from Macintosh systems,
```

```
626 ; or you are running on a Mac and need to deal with files from
627 ; unix or win32 systems, setting this flag will cause PHP to
628 ; automatically detect the EOL character in those files so that
629 ; fgets() and file() will work regardless of the source of the file.
630 ; auto_detect_line_endings = Off
631
632
633 ;;;;;;;;;;;;;;;;;;;;;;;;;;
634 ; Dynamic Extensions ;
635 ;;;;;;;;;;;;;;;;;;;;;;;;;;
636 ;
637 ; If you wish to have an extension loaded automatically, use the following
638 ; syntax:
639 ;
640 ; extension=module.extension
641 ;
642 ; For example, on Windows:
643 ;
644 ; extension=mysqli.dll
645 ;
646 ; ... or under UNIX:
647 ;
648 ; extension=mysqli.so
649 ;
650 ; Note that it should be the name of the module only; no directory information
651 ; needs to go here. Specify the location of the extension with the
652 ; extension_dir directive above.
653
654
655 ; Windows Extensions
656 ; Note that ODBC support is built in, so no dll is needed for it.
657 ; Note that many DLL files are located in the extensions/ (PHP 4) ext/ (PHP 5)
658 ; extension folders as well as the separate PECL DLL download (PHP 5).
659 ; Be sure to appropriately set the extension_dir directive.
660
661 ;extension=php_bz2.dll
662 ;extension=php_curl.dll
663 ;extension=php_dba.dll
664 ;extension=php_dbase.dll
665 ;extension=php_exif.dll
666 ;extension=php_fdf.dll
667 ;extension=php_gd2.dll
668 ;extension=php_gettext.dll
669 ;extension=php_gmp.dll
670 ;extension=php_ifx.dll
671 ;extension=php_imap.dll
672 ;extension=php_interbase.dll
673 ;extension=php_ldap.dll
674 ;extension=php_mbstring.dll
675 ;extension=php_mcrypt.dll
676 ;extension=php_mhash.dll
677 ;extension=php_mime_magic.dll
678 ;extension=php_ming.dll
679 ;extension=php_mysql.dll
680 ;extension=php_mssql.dll
681 ;extension=php_mysql.dll
682 ;extension=php_mysqli.dll
```

```
683 ;extension=php_oci8.dll
684 ;extension=php_openssl.dll
685 ;extension=php_pdo.dll
686 ;extension=php_pdo_firebird.dll
687 ;extension=php_pdo_mssql.dll
688 ;extension=php_pdo_mysql.dll
689 ;extension=php_pdo_oci.dll
690 ;extension=php_pdo_oci8.dll
691 ;extension=php_pdo_odbc.dll
692 ;extension=php_pdo_pgsql.dll
693 ;extension=php_pdo_sqlite.dll
694 ;extension=php_pgsql.dll
695 ;extension=php_pspell.dll
696 ;extension=php_shmop.dll
697 ;extension=php_snmp.dll
698 ;extension=php_soap.dll
699 ;extension=php_sockets.dll
700 ;extension=php_sqlite.dll
701 ;extension=php_sybase_ct.dll
702 ;extension=php_tidy.dll
703 ;extension=php_xmlrpc.dll
704 ;extension=php_xsl.dll
705 ;extension=php_zip.dll
706
707 ;;;;;;;;;;;;;;
708 ; Module Settings ;
709 ;;;;;;;;;;;;;;
710
711 [Date]
712 ; Defines the default timezone used by the date functions
713 ;date.timezone =
714
715 ;date.default_latitude = 31.7667
716 ;date.default_longitude = 35.2333
717
718 ;date.sunrise_zenith = 90.583333
719 ;date.sunset_zenith = 90.583333
720
721 [filter]
722 ;filter.default = unsafe_raw
723 ;filter.default_flags =
724
725 [iconv]
726 ;iconv.input_encoding = ISO-8859-1
727 ;iconv.internal_encoding = ISO-8859-1
728 ;iconv.output_encoding = ISO-8859-1
729
730 [sqlite]
731 ;sqlite.assoc_case = 0
732
733 [Pcre]
734 ;PCRE library backtracking limit.
735 ;pcre.backtrack_limit=100000
736
737 ;PCRE library recursion limit.
738 ;Please note that if you set this value to a high number you may consume all
739 ;the available process stack and eventually crash PHP (due to reaching the
```

```
740 ;stack size limit imposed by the Operating System).
741 ;pcre.recursion_limit=100000
742
743 [Syslog]
744 ; Whether or not to define the various syslog variables (e.g. $LOG_PID,
745 ; $LOG_CRON, etc.). Turning it off is a good idea performance-wise. In
746 ; runtime, you can define these variables by calling define_syslog_variables().
747 define_syslog_variables = Off
748
749 [mail function]
750 ; For Win32 only.
751 SMTP = localhost
752 smtp_port = 25
753
754 ; For Win32 only.
755 ;sendmail_from = me@example.com
756
757 ; For Unix only. You may supply arguments as well (default: "sendmail -t -i").
758 ;sendmail_path =
759
760 ; Force the addition of the specified parameters to be passed as extra parameters
761 ; to the sendmail binary. These parameters will always replace the value of
762 ; the 5th parameter to mail(), even in safe mode.
763 ;mail.force_extra_parameters =
764
765 [SQL]
766 sql.safe_mode = Off
767
768 [ODBC]
769 ;odbc.default_db = Not yet implemented
770 ;odbc.default_user = Not yet implemented
771 ;odbc.default_pw = Not yet implemented
772
773 ; Allow or prevent persistent links.
774 odbc.allow_persistent = On
775
776 ; Check that a connection is still valid before reuse.
777 odbc.check_persistent = On
778
779 ; Maximum number of persistent links. -1 means no limit.
780 odbc.max_persistent = -1
781
782 ; Maximum number of links (persistent + non-persistent). -1 means no limit.
783 odbc.max_links = -1
784
785 ; Handling of LONG fields. Returns number of bytes to variables. 0 means
786 ; passthru.
787 odbc.defaultlrl = 4096
788
789 ; Handling of binary data. 0 means passthru, 1 return as is, 2 convert to char.
790 ; See the documentation on odbc_binmode and odbc_longreadlen for an explanation
791 ; of uodbc.defaultlrl and uodbc.defaultbinmode
792 odbc.defaultbinmode = 1
793
794 [MySQL]
795 ; Allow or prevent persistent links.
796 mysql.allow_persistent = On
```

```
797
798 ; Maximum number of persistent links. -1 means no limit.
799 mysql.max_persistent = -1
800
801 ; Maximum number of links (persistent + non-persistent). -1 means no limit.
802 mysql.max_links = -1
803
804 ; Default port number for mysql_connect(). If unset, mysql_connect() will use
805 ; the $MYSQL_TCP_PORT or the mysql-tcp entry in /etc/services or the
806 ; compile-time value defined MYSQL_PORT (in that order). Win32 will only look
807 ; at MYSQL_PORT.
808 mysql.default_port =
809
810 ; Default socket name for local MySQL connects. If empty, uses the built-in
811 ; MySQL defaults.
812 mysql.default_socket =
813
814 ; Default host for mysql_connect() (doesn't apply in safe mode).
815 mysql.default_host =
816
817 ; Default user for mysql_connect() (doesn't apply in safe mode).
818 mysql.default_user =
819
820 ; Default password for mysql_connect() (doesn't apply in safe mode).
821 ; Note that this is generally a *bad* idea to store passwords in this file.
822 ; *Any* user with PHP access can run 'echo get_cfg_var("mysql.default_password")
823 ; and reveal this password! And of course, any users with read access to this
824 ; file will be able to reveal the password as well.
825 mysql.default_password =
826
827 ; Maximum time (in seconds) for connect timeout. -1 means no limit
828 mysql.connect_timeout = 60
829
830 ; Trace mode. When trace_mode is active (=On), warnings for table/index scans and
831 ; SQL-Errors will be displayed.
832 mysql.trace_mode = Off
833
834 [MySQLi]
835
836 ; Maximum number of links. -1 means no limit.
837 mysqli.max_links = -1
838
839 ; Default port number for mysqli_connect(). If unset, mysqli_connect() will use
840 ; the $MYSQL_TCP_PORT or the mysql-tcp entry in /etc/services or the
841 ; compile-time value defined MYSQL_PORT (in that order). Win32 will only look
842 ; at MYSQL_PORT.
843 mysqli.default_port = 3306
844
845 ; Default socket name for local MySQL connects. If empty, uses the built-in
846 ; MySQL defaults.
847 mysqli.default_socket =
848
849 ; Default host for mysql_connect() (doesn't apply in safe mode).
850 mysqli.default_host =
851
852 ; Default user for mysql_connect() (doesn't apply in safe mode).
853 mysqli.default_user =
```



```
854
855 ; Default password for mysqli_connect() (doesn't apply in safe mode).
856 ; Note that this is generally a *bad* idea to store passwords in this file.
857 ; *Any* user with PHP access can run 'echo get_cfg_var("mysqli.default_pw")
858 ; and reveal this password!  And of course, any users with read access to this
859 ; file will be able to reveal the password as well.
860 mysqli.default_pw =
861
862 ; Allow or prevent reconnect
863 mysqli.reconnect = Off
864
865 [mSQL]
866 ; Allow or prevent persistent links.
867 msql.allow_persistent = On
868
869 ; Maximum number of persistent links.  -1 means no limit.
870 msql.max_persistent = -1
871
872 ; Maximum number of links (persistent+non persistent).  -1 means no limit.
873 msql.max_links = -1
874
875 [OCI8]
876 ; enables privileged connections using external credentials (OCI_SYSOPER, OCI_SYSDBA)
877 ;oci8.privileged_connect = Off
878
879 ; Connection: The maximum number of persistent OCI8 connections per
880 ; process. Using -1 means no limit.
881 ;oci8.max_persistent = -1
882
883 ; Connection: The maximum number of seconds a process is allowed to
884 ; maintain an idle persistent connection. Using -1 means idle
885 ; persistent connections will be maintained forever.
886 ;oci8.persistent_timeout = -1
887
888 ; Connection: The number of seconds that must pass before issuing a
889 ; ping during oci_pconnect() to check the connection validity. When
890 ; set to 0, each oci_pconnect() will cause a ping. Using -1 disables
891 ; pings completely.
892 ;oci8.ping_interval = 60
893
894 ; Tuning: This option enables statement caching, and specifies how
895 ; many statements to cache. Using 0 disables statement caching.
896 ;oci8.statement_cache_size = 20
897
898 ; Tuning: Enables statement prefetching and sets the default number of
899 ; rows that will be fetched automatically after statement execution.
900 ;oci8.default_prefetch = 10
901
902 ; Compatibility. Using On means oci_close() will not close
903 ; oci_connect() and oci_new_connect() connections.
904 ;oci8.old_oci_close_semantics = Off
905
906 [PostgreSQL]
907 ; Allow or prevent persistent links.
908 pgsql.allow_persistent = On
909
910 ; Detect broken persistent links always with pg_pconnect().
```

```
911 ; Auto reset feature requires a little overheads.
912 pgsql.auto_reset_persistent = Off
913
914 ; Maximum number of persistent links. -1 means no limit.
915 pgsql.max_persistent = -1
916
917 ; Maximum number of links (persistent+non persistent). -1 means no limit.
918 pgsql.max_links = -1
919
920 ; Ignore PostgreSQL backends Notice message or not.
921 ; Notice message logging require a little overheads.
922 pgsql.ignore_notice = 0
923
924 ; Log PostgreSQL backends Notice message or not.
925 ; Unless pgsql.ignore_notice=0, module cannot log notice message.
926 pgsql.log_notice = 0
927
928 [Sybase]
929 ; Allow or prevent persistent links.
930 sybase.allow_persistent = On
931
932 ; Maximum number of persistent links. -1 means no limit.
933 sybase.max_persistent = -1
934
935 ; Maximum number of links (persistent + non-persistent). -1 means no limit.
936 sybase.max_links = -1
937
938 ;sybase.interface_file = "/usr/sybase/interfaces"
939
940 ; Minimum error severity to display.
941 sybase.min_error_severity = 10
942
943 ; Minimum message severity to display.
944 sybase.min_message_severity = 10
945
946 ; Compatibility mode with old versions of PHP 3.0.
947 ; If on, this will cause PHP to automatically assign types to results according
948 ; to their Sybase type, instead of treating them all as strings. This
949 ; compatibility mode will probably not stay around forever, so try applying
950 ; whatever necessary changes to your code, and turn it off.
951 sybase.compatibility_mode = Off
952
953 [Sybase-CT]
954 ; Allow or prevent persistent links.
955 sybct.allow_persistent = On
956
957 ; Maximum number of persistent links. -1 means no limit.
958 sybct.max_persistent = -1
959
960 ; Maximum number of links (persistent + non-persistent). -1 means no limit.
961 sybct.max_links = -1
962
963 ; Minimum server message severity to display.
964 sybct.min_server_severity = 10
965
966 ; Minimum client message severity to display.
967 sybct.min_client_severity = 10
```

```
968
969 [bcmath]
970 ; Number of decimal digits for all bcmath functions.
971 bcmath.scale = 0
972
973 [browscap]
974 ;browscap = extra/browscap.ini
975
976 [Informix]
977 ; Default host for ifx_connect() (doesn't apply in safe mode).
978 ifx.default_host =
979
980 ; Default user for ifx_connect() (doesn't apply in safe mode).
981 ifx.default_user =
982
983 ; Default password for ifx_connect() (doesn't apply in safe mode).
984 ifx.default_password =
985
986 ; Allow or prevent persistent links.
987 ifx.allow_persistent = On
988
989 ; Maximum number of persistent links. -1 means no limit.
990 ifx.max_persistent = -1
991
992 ; Maximum number of links (persistent + non-persistent). -1 means no limit.
993 ifx.max_links = -1
994
995 ; If on, select statements return the contents of a text blob instead of its id.
996 ifx.textasvarchar = 0
997
998 ; If on, select statements return the contents of a byte blob instead of its id.
999 ifx.byteasvarchar = 0
1000
1001 ; Trailing blanks are stripped from fixed-length char columns. May help the
1002 ; life of Informix SE users.
1003 ifx.charasvarchar = 0
1004
1005 ; If on, the contents of text and byte blobs are dumped to a file instead of
1006 ; keeping them in memory.
1007 ifx.blobinfile = 0
1008
1009 ; NULL's are returned as empty strings, unless this is set to 1. In that case,
1010 ; NULL's are returned as string 'NULL'.
1011 ifx.nullformat = 0
1012
1013 [Session]
1014 ; Handler used to store/retrieve data.
1015 session.save_handler = files
1016
1017 ; Argument passed to save_handler. In the case of files, this is the path
1018 ; where data files are stored. Note: Windows users have to change this
1019 ; variable in order to use PHP's session functions.
1020 ;
1021 ; As of PHP 4.0.1, you can define the path as:
1022 ;
1023 ;     session.save_path = "N;/path"
1024 ;
```

```
1025 ; where N is an integer. Instead of storing all the session files in
1026 ; /path, what this will do is use subdirectories N-levels deep, and
1027 ; store the session data in those directories. This is useful if you
1028 ; or your OS have problems with lots of files in one directory, and is
1029 ; a more efficient layout for servers that handle lots of sessions.
1030 ;
1031 ; NOTE 1: PHP will not create this directory structure automatically.
1032 ;         You can use the script in the ext/session dir for that purpose.
1033 ; NOTE 2: See the section on garbage collection below if you choose to
1034 ;         use subdirectories for session storage
1035 ;
1036 ; The file storage module creates files using mode 600 by default.
1037 ; You can change that by using
1038 ;
1039 ;     session.save_path = "N;MODE;/path"
1040 ;
1041 ; where MODE is the octal representation of the mode. Note that this
1042 ; does not overwrite the process's umask.
1043 ;session.save_path = "/tmp"
1044
1045 ; Whether to use cookies.
1046 session.use_cookies = 1
1047
1048 ;session.cookie_secure =
1049
1050 ; This option enables administrators to make their users invulnerable to
1051 ; attacks which involve passing session ids in URLs; defaults to 0.
1052 ; session.use_only_cookies = 1
1053
1054 ; Name of the session (used as cookie name).
1055 session.name = PHPSESSID
1056
1057 ; Initialize session on request startup.
1058 session.auto_start = 0
1059
1060 ; Lifetime in seconds of cookie or, if 0, until browser is restarted.
1061 session.cookie_lifetime = 0
1062
1063 ; The path for which the cookie is valid.
1064 session.cookie_path = /
1065
1066 ; The domain for which the cookie is valid.
1067 session.cookie_domain =
1068
1069 ; Whether or not to add the httpOnly flag to the cookie, which makes it inaccessible
1070 ; to browser scripting languages such as JavaScript.
1071 session.cookie_httponly =
1072
1073 ; Handler used to serialize data. php is the standard serializer of PHP.
1074 session.serialize_handler = php
1075
1076 ; Define the probability that the 'garbage collection' process is started
1077 ; on every session initialization.
1078 ; The probability is calculated by using gc_probability/gc_divisor,
1079 ; e.g. 1/100 means there is a 1% chance that the GC process starts
1080 ; on each request.
```

```
1081 session.gc_probability = 1
1082 session.gc_divisor = 1000
1083
1084 ; After this number of seconds, stored data will be seen as 'garbage' and
1085 ; cleaned up by the garbage collection process.
1086 session.gc_maxlifetime = 1440
1087
1088 ; NOTE: If you are using the subdirectory option for storing session files
1089 ; (see session.save_path above), then garbage collection does *not*
1090 ; happen automatically. You will need to do your own garbage
1091 ; collection through a shell script, cron entry, or some other method.
1092 ; For example, the following script would is the equivalent of
1093 ; setting session.gc_maxlifetime to 1440 (1440 seconds = 24 minutes):
1094 ;     cd /path/to/sessions; find -cmin +24 | xargs rm
1095
1096 ; PHP 4.2 and less have an undocumented feature/bug that allows you to
1097 ; to initialize a session variable in the global scope, albeit register_globals
1098 ; is disabled. PHP 4.3 and later will warn you, if this feature is used.
1099 ; You can disable the feature and the warning separately. At this time,
1100 ; the warning is only displayed, if bug_compat_42 is enabled.
1101
1102 session.bug_compat_42 = 0
1103 session.bug_compat_warn = 1
1104
1105 ; Check HTTP Referer to invalidate externally stored URLs containing ids.
1106 ; HTTP_REFERER has to contain this substring for the session to be
1107 ; considered as valid.
1108 session.referer_check =
1109
1110 ; How many bytes to read from the file.
1111 session.entropy_length = 0
1112
1113 ; Specified here to create the session id.
1114 session.entropy_file =
1115
1116 ;session.entropy_length = 16
1117
1118 ;session.entropy_file = /dev/urandom
1119
1120 ; Set to {nocache,private,public,} to determine HTTP caching aspects
1121 ; or leave this empty to avoid sending anti-caching headers.
1122 session.cache_limiter = nocache
1123
1124 ; Document expires after n minutes.
1125 session.cache_expire = 180
1126
1127 ; trans sid support is disabled by default.
1128 ; Use of trans sid may risk your users security.
1129 ; Use this option with caution.
1130 ; - User may send URL contains active session ID
1131 ;   to other person via. email/irc/etc.
1132 ; - URL that contains active session ID may be stored
1133 ;   in publically accessible computer.
1134 ; - User may access your site with the same session ID
1135 ;   always using URL stored in browser's history or bookmarks.
1136 session.use_trans_sid = 0
1137
```

```
1138 ; Select a hash function
1139 ; 0: MD5 (128 bits)
1140 ; 1: SHA-1 (160 bits)
1141 session.hash_function = 0
1142
1143 ; Define how many bits are stored in each character when converting
1144 ; the binary hash data to something readable.
1145 ;
1146 ; 4 bits: 0-9, a-f
1147 ; 5 bits: 0-9, a-v
1148 ; 6 bits: 0-9, a-z, A-Z, "-", ",", "
1149 session.hash_bits_per_character = 5
1150
1151 ; The URL rewriter will look for URLs in a defined set of HTML tags.
1152 ; form/fieldset are special; if you include them here, the rewriter will
1153 ; add a hidden <input> field with the info which is otherwise appended
1154 ; to URLs. If you want XHTML conformity, remove the form entry.
1155 ; Note that all valid entries require a "=", even if no value follows.
1156 url_rewriter.tags = "a:href,area:href,frame=src,input=src,form=fakeentry"
1157
1158 [MSSQL]
1159 ; Allow or prevent persistent links.
1160 mssql.allow_persistent = On
1161
1162 ; Maximum number of persistent links. -1 means no limit.
1163 mssql.max_persistent = -1
1164
1165 ; Maximum number of links (persistent+non persistent). -1 means no limit.
1166 mssql.max_links = -1
1167
1168 ; Minimum error severity to display.
1169 mssql.min_error_severity = 10
1170
1171 ; Minimum message severity to display.
1172 mssql.min_message_severity = 10
1173
1174 ; Compatibility mode with old versions of PHP 3.0.
1175 mssql.compatability_mode = Off
1176
1177 ; Connect timeout
1178 ;mssql.connect_timeout = 5
1179
1180 ; Query timeout
1181 ;mssql.timeout = 60
1182
1183 ; Valid range 0 - 2147483647. Default = 4096.
1184 ;mssql.textlimit = 4096
1185
1186 ; Valid range 0 - 2147483647. Default = 4096.
1187 ;mssql.textsize = 4096
1188
1189 ; Limits the number of records in each batch. 0 = all records in one batch.
1190 ;mssql.batchsize = 0
1191
1192 ; Specify how datetime and datetim4 columns are returned
1193 ; On => Returns data converted to SQL server settings
1194 ; Off => Returns values as YYYY-MM-DD hh:mm:ss
```

```
1195 ;mssql.datetimeconvert = On
1196
1197 ; Use NT authentication when connecting to the server
1198 mssql.secure_connection = Off
1199
1200 ; Specify max number of processes. -1 = library default
1201 ; msdlib defaults to 25
1202 ; FreeTDS defaults to 4096
1203 ;mssql.max_procs = -1
1204
1205 ; Specify client character set.
1206 ; If empty or not set the client charset from freetds.comf is used
1207 ; This is only used when compiled with FreeTDS
1208 ;mssql.charset = "ISO-8859-1"
1209
1210 [Assertion]
1211 ; Assert(expr); active by default.
1212 ;assert.active = On
1213
1214 ; Issue a PHP warning for each failed assertion.
1215 ;assert.warning = On
1216
1217 ; Don't bail out by default.
1218 ;assert.bail = Off
1219
1220 ; User-function to be called if an assertion fails.
1221 ;assert.callback = 0
1222
1223 ; Eval the expression with current error_reporting(). Set to true if you want
1224 ; error_reporting(0) around the eval().
1225 ;assert.quiet_eval = 0
1226
1227 [COM]
1228 ; path to a file containing GUIDs, IIDs or filenames of files with TypeLibs
1229 ;com.typelib_file =
1230 ; allow Distributed-COM calls
1231 ;com.allow_dcom = true
1232 ; autoregister constants of a components typlib on com_load()
1233 ;com.autoregister_typelib = true
1234 ; register constants casesensitive
1235 ;com.autoregister_casesensitive = false
1236 ; show warnings on duplicate constant registrations
1237 ;com.autoregister_verbose = true
1238
1239 [mbstring]
1240 ; language for internal character representation.
1241 ;mbstring.language = Japanese
1242
1243 ; internal/script encoding.
1244 ; Some encoding cannot work as internal encoding.
1245 ; (e.g. SJIS, BIG5, ISO-2022-*)
1246 ;mbstring.internal_encoding = EUJ-JP
1247
1248 ; http input encoding.
1249 ;mbstring.http_input = auto
1250
1251 ; http output encoding. mb_output_handler must be
```



```
1252 ; registered as output buffer to function
1253 ;mbstring.http_output = SJIS
1254
1255 ; enable automatic encoding translation according to
1256 ; mbstring.internal_encoding setting. Input chars are
1257 ; converted to internal encoding by setting this to On.
1258 ; Note: Do not use automatic encoding translation for
1259 ;     portable libs/applications.
1260 ;mbstring.encoding_translation = Off
1261
1262 ; automatic encoding detection order.
1263 ; auto means
1264 ;mbstring.detect_order = auto
1265
1266 ; substitute_character used when character cannot be converted
1267 ; one from another
1268 ;mbstring.substitute_character = none;
1269
1270 ; overload(replace) single byte functions by mbstring functions.
1271 ; mail(), ereg(), etc are overloaded by mb_send_mail(), mb_ereg(),
1272 ; etc. Possible values are 0,1,2,4 or combination of them.
1273 ; For example, 7 for overload everything.
1274 ; 0: No overload
1275 ; 1: Overload mail() function
1276 ; 2: Overload str*() functions
1277 ; 4: Overload ereg*() functions
1278 ;mbstring.func_overload = 0
1279
1280 ; enable strict encoding detection.
1281 ;mbstring.strict_encoding = Off
1282
1283 [FrontBase]
1284 ;fbsql.allow_persistent = On
1285 ;fbsql.autocommit = On
1286 ;fbsql.show_timestamp_decimals = Off
1287 ;fbsql.default_database =
1288 ;fbsql.default_database_password =
1289 ;fbsql.default_host =
1290 ;fbsql.default_password =
1291 ;fbsql.default_user = "_SYSTEM"
1292 ;fbsql.generate_warnings = Off
1293 ;fbsql.max_connections = 128
1294 ;fbsql.max_links = 128
1295 ;fbsql.max_persistent = -1
1296 ;fbsql.max_results = 128
1297
1298 [gd]
1299 ; Tell the jpeg decode to libjpeg warnings and try to create
1300 ; a gd image. The warning will then be displayed as notices
1301 ; disabled by default
1302 ;gd.jpeg_ignore_warning = 0
1303
1304 [exif]
1305 ; Exif UNICODE user comments are handled as UCS-2BE/UCS-2LE and JIS as JIS.
1306 ; With mbstring support this will automatically be converted into the encoding
1307 ; given by corresponding encode setting. When empty mbstring.internal_encoding
1308 ; is used. For the decode settings you can distinguish between motorola and
```

```
1309 ; intel byte order. A decode setting cannot be empty.
1310 ;exif.encode_unicode = ISO-8859-15
1311 ;exif.decode_unicode_motorola = UCS-2BE
1312 ;exif.decode_unicode_intel    = UCS-2LE
1313 ;exif.encode_jis =
1314 ;exif.decode_jis_motorola = JIS
1315 ;exif.decode_jis_intel    = JIS
1316
1317 [Tidy]
1318 ; The path to a default tidy configuration file to use when using tidy
1319 ;tidy.default_config = /usr/local/lib/php/default.tcfg
1320
1321 ; Should tidy clean and repair output automatically?
1322 ; WARNING: Do not use this option if you are generating non-html content
1323 ; such as dynamic images
1324 tidy.clean_output = Off
1325
1326 [soap]
1327 ; Enables or disables WSDL caching feature.
1328 soap.wsdl_cache_enabled=1
1329 ; Sets the directory name where SOAP extension will put cache files.
1330 soap.wsdl_cache_dir="/tmp"
1331 ; (time to live) Sets the number of second while cached file will be used
1332 ; instead of original one.
1333 soap.wsdl_cache_ttl=86400
1334
1335 ; Local Variables:
1336 ; tab-width: 4
1337 ; End:
1338
```