Apache

Introduction

These functions are only available when running PHP as an Apache module.

Note
As of PHP 4.3.2, PATH_TRANSLATED is no longer set implicitly under the Apache 2 SAPI in contrast to the situation in Apache 1, where it's set to the same value as the SCRIPT_FILENAME server variable when it's not populated by Apache. This change was made to comply with the CGI specification that PATH_TRANSLATED should only exist if PATH_INFO is defined. Apache 2 users may use *AcceptPathInfo = On* inside *httpd.conf* to define PATH_INFO.

Installing/Configuring

Requirements

No external libraries are needed to build this extension.

Installation

For PHP installation on Apache see the installation chapter.

Runtime Configuration

The behaviour of the Apache PHP module is affected by settings in *php.ini*. Configuration settings from *php.ini* may be overridden by php_flag settings in the server configuration file or local*.htaccess* files.

Example #1 - Turning off PHP parsing for a directory using*.htaccess*

```
php_flag engine off
```

Apache configuration options

Name	Default	Changeable	Changelog
engine	"1"	PHP_INI_ALL	Available since PHP 4.0.5.
child_terminate	"0"	PHP_INI_ALL	Available since PHP 4.0.5.
last_modified	"0"	PHP_INI_ALL	Available since PHP 4.0.5.
xbithack	"0"	PHP_INI_ALL	Available since PHP 4.0.5.

For further details and definitions of the PHP_INI_* constants, see the php.ini directives.

Here's a short explanation of the configuration directives.

engine boolean

Turns PHP parsing on or off. This directive is really only useful in the Apache module version of PHP. It is used by sites that would like to turn PHP parsing on and off on a per-directory or per-virtual server basis. By putting **engine off** in the appropriate places in the *httpd.conf* file, PHP can be enabled or disabled.

`child_terminate` boolean
Specify whether PHP scripts may request child process termination on end of request, see also apache_child_terminate().

`last_modified` boolean
Send PHP scripts modification date as Last-Modified: header for this request.

`xbithack` boolean
Parse files with executable bit set as PHP regardless of their file ending.

Resource Types

This extension has no resource types defined.

Predefined Constants

This extension has no constants defined.

Apache Functions

apache_child_terminate

apache_child_terminate -- Terminate apache process after this request

Description

bool **apache_child_terminate** (void)

apache_child_terminate() will register the Apache process executing the current PHP request for termination once execution of PHP code is completed. It may be used to terminate a process after a script with high memory consumption has been run as memory will usually only be freed internally but not given back to the operating system.

Return Values

Returns **TRUE** if PHP is running as an Apache 1 module, the Apache version is non-multithreaded, and the child_terminate PHP directive is enabled (disabled by default). If these conditions are not met, **FALSE** is returned and an error of level **E_WARNING** is generated.

Notes

Note
This function is not implemented on Windows platforms.

See Also

- exit()

apache_get_modules

apache_get_modules -- Get a list of loaded Apache modules

Description

array **apache_get_modules** (void)

Get a list of loaded Apache modules.

Return Values

An array of loaded Apache modules.

ChangeLog

Version	Description
5.0.0	Became available when using Apache 1, or the PHP Apache 2 *filter* API. Before this time, it was only available when using the Apache 2 *handler* API.

Examples

Example #2 - apache_get_modules() example

```
<?php
print_r(apache_get_modules());
?>
```

The above example will output something similar to:

```
Array
(
    [0] => core
    [1] => http_core
    [2] => mod_so
    [3] => sapi_apache2
    [4] => mod_mime
    [5] => mod_rewrite
)
```

apache_get_version

apache_get_version -- Fetch Apache version

Description

string **apache_get_version** (void)

Fetch the Apache version.

Return Values

Returns the Apache version on success, or **FALSE** on failure.

ChangeLog

Version	Description
4.3.4	Became available with Apache 1.
5.0.0	Became available with the Apache 2 *filter* API.

Examples

Example #3 - apache_get_version() example

```
<?php
$version = apache_get_version();
echo "$version\n";
?>
```

The above example will output something similar to:

```
Apache/1.3.29 (Unix) PHP/4.3.4
```

See Also

- phpinfo()

apache_getenv

apache_getenv -- Get an Apache subprocess_env variable

Description

string **apache_getenv** (string `$variable` [, bool `$walk_to_top`])

Get an Apache environment variable as specified by *`variable`*.

This function requires Apache 2 otherwise it's undefined.

Parameters

`variable`
: The Apache environment variable

`walk_to_top`
: Whether to get the top-level variable available to all Apache layers.

Return Values

The value of the Apache environment variable on success, or **FALSE** on failure

Examples

Example #4 - apache_getenv() example

The example above shows how to retrieve the value of the Apache environment variable *SERVER_ADDR*.

```
<?php
$ret = apache_getenv("SERVER_ADDR");
echo $ret;
?>
```

The above example will output something similar to:

```
42.24.42.240
```

See Also

- apache_setenv()
- getenv()
- Superglobals

apache_lookup_uri

apache_lookup_uri -- Perform a partial request for the specified URI and return all info about it

Description

object **apache_lookup_uri** (string `$filename`)

This performs a partial request for a URI. It goes just far enough to obtain all the important information about the given resource.

This function is only supported when PHP is installed as an Apache module.

Parameters

`filename`
The filename (URI) that's being requested.

Return Values

An object of related URI information. The properties of this object are:

- 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

Examples

Example #5 - apache_lookup_uri() example

```
<?php
$info = apache_lookup_uri('index.php?var=value');
print_r($info);

if (file_exists($info->filename)) {
   echo 'file exists!';
}
?>
```

The above example will output something similar to:

```
stdClass Object
(
   [status] => 200
   [the_request] => GET /dir/file.php HTTP/1.1
   [method] => GET
   [mtime] => 0
   [clength] => 0
   [chunked] => 0
   [content_type] => application/x-httpd-php
   [no_cache] => 0
   [no_local_copy] => 1
   [unparsed_uri] => /dir/index.php?var=value
   [uri] => /dir/index.php
   [filename] => /home/htdocs/dir/index.php
   [args] => var=value
   [allowed] => 0
   [sent_bodyct] => 0
   [bytes_sent] => 0
   [request_time] => 1074282764
)
file exists!
```

apache_note

apache_note -- Get and set apache request notes

Description

string **apache_note** (string `$note_name` [, string `$note_value`])

This function is a wrapper for Apache's *table_get* and *table_set*. It edits the table of notes that exists during a request. The table's purpose is to allow Apache modules to communicate.

The main use for apache_note() is to pass information from one module to another within the same request.

Parameters

`note_name`
The name of the note.

`note_value`
The value of the note.

Return Values

If called with one argument, it returns the current value of note *note_name*. If called with two arguments, it sets the value of note *note_name* to *note_value* and returns the previous value of note *note_name*. If the note cannot be retrieved, **FALSE** is returned.

Examples

Example #6 - Passing information between PHP and Perl

```
<?php

apache_note('name', 'Fredrik Ekengren');

// Call perl script
virtual("/perl/some_script.pl");

$result = apache_note("resultdata");
?>

# Get Apache request object
my $r = Apache->request()->main();
```

```
# Get passed data
my $name = $r->notes('name');

# some processing

# Pass result back to PHP
$r->notes('resultdata', $result);
```

Example #7 - Logging values in access.log

```
<?php

apache_note('sessionID', session_id());

?>

# "%{sessionID}n" can be used in the LogFormat directive
```

See Also

- virtual()

apache_request_headers

apache_request_headers -- Fetch all HTTP request headers

Description

array **apache_request_headers** (void)

Fetches all HTTP requests from the current request.

This function is only supported when PHP is installed as an Apache module.

Return Values

An associative array of all the HTTP headers in the current request, or **FALSE** on failure.

Examples

Example #8 - apache_request_headers() **example**

```
<?php
$headers = apache_request_headers();

foreach ($headers as $header => $value) {
   echo "$header: $value <br />\n";
}
?>
```

The above example will output something similar to:

```
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Host: www.example.com
Connection: Keep-Alive
```

Notes

Note

Prior to PHP 4.3.0, apache_request_headers() was called getallheaders(). After PHP 4.3.0, getallheaders() is an alias for apache_request_headers().

Note

You can also get at the value of the common CGI variables by reading them from the environment, which works whether or not you are using PHP as an Apache module. Use phpinfo() to see a list of all of the available environment variables.

Note

As of PHP 4.3.3 you can use this function with the NSAPI server module in Netscape/iPlanet/SunONE webservers, too.

See Also

- apache_response_headers()

apache_reset_timeout

apache_reset_timeout -- Reset the Apache write timer

Description

bool **apache_reset_timeout** (void)

apache_reset_timeout() resets the Apache write timer, which defaults to 300 seconds. With *set_time_limit(0); ignore_user_abort(true)* and periodic apache_reset_timeout() calls, Apache can theoretically run forever.

This function requires Apache 1.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Notes

Note
This function is disabled when PHP is running in safe mode.

See Also

- set_time_limit()
- ignore_user_abort()

apache_response_headers

apache_response_headers -- Fetch all HTTP response headers

Description

array **apache_response_headers** (void)

Fetch all HTTP response headers.

Return Values

An array of all Apache response headers on success, or **FALSE** on failure.

Examples

Example #9 - apache_response_headers() **example**

```
<?php
print_r(apache_response_headers());
?>
```

The above example will output something similar to:

```
Array
(
   [Accept-Ranges] => bytes
   [X-Powered-By] => PHP/4.3.8
)
```

Notes

Note

As of PHP 4.3.3 you can use this function with the NSAPI server module in Netscape/iPlanet/SunONE webservers, too.

See Also

- apache_request_headers()
- headers_sent()
- headers_list()

apache_setenv

apache_setenv -- Set an Apache subprocess_env variable

Description

bool **apache_setenv** (string `$variable`, string `$value` [, bool `$walk_to_top`])

apache_setenv() sets the value of the Apache environment variable specified by *`variable`* .

Note
When setting an Apache environment variable, the corresponding $_SERVER variable is not changed.

Parameters

`variable`
: The environment variable that's being set.

`value`
: The new *`variable`* value.

`walk_to_top`
: Whether to set the top-level variable available to all Apache layers.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #10 - Setting an Apache environment variable using apache_setenv()

```
<?php
apache_setenv("EXAMPLE_VAR", "Example Value");
?>
```

Notes

Note
apache_setenv() can be paired up with apache_getenv() across separate pages or for setting variables to pass to Server Side Includes (.shtml) that have been included in PHP scripts.

See Also

- apache_getenv()

ascii2ebcdic

ascii2ebcdic -- Translate string from ASCII to EBCDIC

Description

int **ascii2ebcdic** (string `$ascii_str`)

ascii2ebcdic() is an Apache-specific function which is available only on EBCDIC based operating systems (OS/390, BS2000). It translates the ASCII encoded string *`ascii_str`* to its equivalent EBCDIC representation (binary safe), and returns the result.

Parameters

`ascii_str`
The ASCII string that will be translated.

Return Values

The EBCDIC representation of an ASCII string.

See Also

- ebcdic2ascii()

ebcdic2ascii

ebcdic2ascii -- Translate string from EBCDIC to ASCII

Description

int **ebcdic2ascii** (string `$ebcdic_str`)

ebcdic2ascii() is an Apache-specific function which is available only on EBCDIC based operating systems (OS/390, BS2000). It translates the EBCDIC encoded string *`ebcdic_str`* to its equivalent ASCII representation (binary safe), and returns the result.

Parameters

`ebcdic_str`
The EBCDIC string that will be translated.

Return Values

The ASCII representation of an EBCDIC string.

See Also

- ascii2ebcdic()

getallheaders

getallheaders -- Fetch all HTTP request headers

Description

array **getallheaders** (void)

Fetches all HTTP headers from the current request.

This function is an alias for apache_request_headers(). Please read the apache_request_headers() documentation for more information on how this function works.

This function is only supported when PHP is installed as an Apache module.

Return Values

An associative array of all the HTTP headers in the current request, or **FALSE** on failure.

ChangeLog

Version	Description
4.3.0	Became an alias for apache_request_headers(). Essentially, it was renamed. This is because this function only works with Apache.

Examples

Example #11 - getallheadres() example

```
<?php

foreach (getallheaders() as $name => $value) {
   echo "$name: $value\n";
}

?>
```

Notes

Note
As of PHP 4.3.3 you can use this function with the NSAPI server module in Netscape/iPlanet/SunONE webservers, too.

See Also

- apache_response_headers()

virtual

virtual -- Perform an Apache sub-request

Description

bool **virtual** (string `$filename`)

virtual() is an Apache-specific function which is similar to *<!--#include virtual...-->* in mod_include. It performs an Apache sub-request. It is useful for including CGI scripts or .shtml files, or anything else that you would parse through Apache. Note that for a CGI script, the script must generate valid CGI headers. At the minimum that means it must generate a Content-type header.

To run the sub-request, all buffers are terminated and flushed to the browser, pending headers are sent too.

This function is only supported when PHP is installed as an Apache module.

Parameters

`filename`
The file that the virtual command will be performed on.

Return Values

Performs the virtual command on success, or returns **FALSE** on failure.

ChangeLog

Version	Description
4.0.6	This function may be used on PHP files. However, it is typically better to use **include()** or **require()** for PHP files.

Examples

See apache_note() for an example.

Notes

Warning
The query string can be passed to the included file but *$_GET* is copied from the parent script and only *$_SERVER['QUERY_STRING']* is filled with the passed query string. The query string may only be passed when using Apache 2. The requested file will not be listed in the Apache access log.

Note
Environment variables set in the requested file are not visible to the calling script.

Note
As of PHP 4.3.3 you can use this function with the NSAPI server module in Netscape/iPlanet/SunONE webservers, too.

See Also

- apache_note()