# $_SERVER

(PHP 4 >= 4.1.0, PHP 5, PHP 7, PHP 8)

$_SERVER — Server and execution environment information

## Description ¶

*$_SERVER* is an array containing information such as headers, paths, and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these; servers may omit some, or provide others not listed here. That said, a large number of these variables are accounted for in the » CGI/1.1 specification, so you should be able to expect those.

## Indices ¶

You may or may not find any of the following elements in *$_SERVER*. Note that few, if any, of these will be available (or indeed have any meaning) if running PHP on the command line.

'*PHP_SELF*'
: The filename of the currently executing script, relative to the document root. For instance,*$_SERVER['PHP_SELF']* in a script at the address *http://example.com/foo/bar.php* would be */foo/bar.php*. The __FILE__ constant contains the full path and filename of the current (i.e. included) file. If PHP is running as a command-line processor this variable contains the script name.

'argv'
: Array of arguments passed to the script. When the script is run on the command line, this gives C-style access to the command line parameters. When called via the GET method, this will contain the query string.

'argc'
: Contains the number of command line parameters passed to the script (if run on the command line).

'*GATEWAY_INTERFACE*'
: What revision of the CGI specification the server is using; e.g. '`CGI/1.1`'.

'*SERVER_ADDR*'
: The IP address of the server under which the current script is executing.

'*SERVER_NAME*'
: The name of the server host under which the current script is executing. If the script is running on a virtual host, this will be the value defined for that virtual host.

> **Note**: Under Apache 2, you must set `UseCanonicalName = On` and `ServerName`. Otherwise, this value reflects the hostname supplied by the client, which can be spoofed. It is not safe to rely on this value in security-dependent contexts.

'*SERVER_SOFTWARE*'
: Server identification string, given in the headers when responding to requests.

'*SERVER_PROTOCOL*'
: Name and revision of the information protocol via which the page was requested; e.g. '`HTTP/1.0`';

'*REQUEST_METHOD*'
: Which request method was used to access the page; e.g. '`GET`', '`HEAD`', '`POST`', '`PUT`'.

> **Note**:
>
> PHP script is terminated after sending headers (it means after producing any output without output buffering) if the request method was `HEAD`.

'*REQUEST_TIME*'
: The timestamp of the start of the request.

'*REQUEST_TIME_FLOAT*'

The timestamp of the start of the request, with microsecond precision.

'*QUERY_STRING*'
The query string, if any, via which the page was accessed.

'*DOCUMENT_ROOT*'
The document root directory under which the current script is executing, as defined in the server's configuration file.

'*HTTP_ACCEPT*'
Contents of the `Accept:` header from the current request, if there is one.

'*HTTP_ACCEPT_CHARSET*'
Contents of the `Accept-Charset:` header from the current request, if there is one. Example: '`iso-8859-1,*,utf-8`'.

'*HTTP_ACCEPT_ENCODING*'
Contents of the `Accept-Encoding:` header from the current request, if there is one. Example: '`gzip`'.

'*HTTP_ACCEPT_LANGUAGE*'
Contents of the `Accept-Language:` header from the current request, if there is one. Example: '`en`'.

'*HTTP_CONNECTION*'
Contents of the `Connection:` header from the current request, if there is one. Example: '`Keep-Alive`'.

'*HTTP_HOST*'
Contents of the `Host:` header from the current request, if there is one.

'*HTTP_REFERER*'
The address of the page (if any) which referred the user agent to the current page. This is set by the user agent. Not all user agents will set this, and some provide the ability to modify *HTTP_REFERER*as a feature. In short, it cannot really be trusted.

'*HTTP_USER_AGENT*'
Contents of the `User-Agent:` header from the current request, if there is one. This is a string denoting the user agent being which is accessing the page. A typical example is: Mozilla/4.5 [en] (X11; U; Linux 2.2.9 i586). Among other things, you can use this value with get_browser() to tailor your page's output to the capabilities of the user agent.

'*HTTPS*'
Set to a non-empty value if the script was queried through the HTTPS protocol.

'*REMOTE_ADDR*'
The IP address from which the user is viewing the current page.

'*REMOTE_HOST*'
The Host name from which the user is viewing the current page. The reverse dns lookup is based on the *REMOTE_ADDR* of the user.

> **Note**: Your web server must be configured to create this variable. For example in Apache you'll need `HostnameLookups On` inside *httpd.conf* for it to exist. See also gethostbyaddr().

'*REMOTE_PORT*'
The port being used on the user's machine to communicate with the web server.

'*REMOTE_USER*'
The authenticated user.

'*REDIRECT_REMOTE_USER*'
The authenticated user if the request is internally redirected.

'*SCRIPT_FILENAME*'

The absolute pathname of the currently executing script.

> **Note**:
>
> If a script is executed with the CLI, as a relative path, such as *file.php* or *../file.php*,*$_SERVER['SCRIPT_FILENAME']* will contain the relative path specified by the user.

'*SERVER_ADMIN*'
The value given to the SERVER_ADMIN (for Apache) directive in the web server configuration file. If the script is running on a virtual host, this will be the value defined for that virtual host.

'*SERVER_PORT*'
The port on the server machine being used by the web server for communication. For default setups, this will be '`80`'; using SSL, for instance, will change this to whatever your defined secure HTTP port is.

> **Note**: Under the Apache 2, you must set `UseCanonicalName = On`, as well as `UseCanonicalPhysicalPort = On` in order to get the physical (real) port, otherwise, this value can be spoofed and it may or may not return the physical port value. It is not safe to rely on this value in security-dependent contexts.

'*SERVER_SIGNATURE*'
String containing the server version and virtual host name which are added to server-generated pages, if enabled.

'*PATH_TRANSLATED*'
Filesystem- (not document root-) based path to the current script, after the server has done any virtual-to-real mapping.

> **Note**: Apache 2 users may use `AcceptPathInfo = On` inside *httpd.conf* to define *PATH_INFO*.

'*SCRIPT_NAME*'
Contains the current script's path. This is useful for pages which need to point to themselves. The __FILE__ constant contains the full path and filename of the current (i.e. included) file.

'*REQUEST_URI*'
The URI which was given in order to access this page; for instance, '`/index.html`'.

'*PHP_AUTH_DIGEST*'
When doing Digest HTTP authentication this variable is set to the 'Authorization' header sent by the client (which you should then use to make the appropriate validation).

'*PHP_AUTH_USER*'
When doing HTTP authentication this variable is set to the username provided by the user.

'*PHP_AUTH_PW*'
When doing HTTP authentication this variable is set to the password provided by the user.

'*AUTH_TYPE*'
When doing HTTP authentication this variable is set to the authentication type.

'*PATH_INFO*'
Contains any client-provided pathname information trailing the actual script filename but preceding the query string, if available. For instance, if the current script was accessed via the URL *http://www.example.com/php/path_info.php/some/stuff?foo=bar*, then *$_SERVER['PATH_INFO']*would contain `/some/stuff`.

'*ORIG_PATH_INFO*'
Original version of '*PATH_INFO*' before processed by PHP.

## Examples ¶

### Example #1 *$_SERVER* example

```
<?php
echo $_SERVER['SERVER_NAME'];
?>
```

The above example will output something similar to:

```
www.example.com
```

## Notes ¶

> **Note**:

This is a 'superglobal', or automatic global, variable. This simply means that it is available in all scopes throughout a script. There is no need to do **global $variable;** to access it within functions or methods.