

## About PHPsuexec

Author:  
eMax Hosting, LLC

Created On: 07 Dec 2005 10:17 PM

---

The PHP installation on our servers runs in CGI PHPsuexec mode. This article provides information about this mode and details the differences you may encounter between PHP running in CGI mode and PHP running as an Apache module.

### PHPSuexec Information

#### File/Directory Permissions

When PHP runs as an Apache Module it executes as the user/group of the webserver which is usually "nobody". Under this mode, files or directories that you require your php scripts to write to need to have 777 permissions (read/write/execute at user/group/world level). This is not very secure because besides allowing the webserver to write to the file it also allows anyone else to read or write to the file.

With PHP running as CGI with suexec enabled your php scripts now execute under your user/group level. Files or directories that you require your php scripts to write to no longer need to have 777 permissions. In fact, having 777 permissions on your scripts or the directories they reside in will not run and will instead cause a 500 internal server error when attempting to execute them to protect you from someone abusing your scripts. Your scripts and directories can have a maximum of 755 permissions (read/write/execute by you, read/execute by everyone else). PHP running as CGI/suexec is much more secure than the older Apache module method.

Files and directories also need to be owned by your user/group. You probably don't need to worry about this as all files you upload or create will be owned by your user/group automatically.

#### .htaccess

Under the old Apache Module mode you were able to manipulate the PHP settings from within a .htaccess file placed in the script's directory.

For example you could turn off the php setting "magic\_quotes\_gpc" with this line in .htaccess:

```
php_value magic_quotes_gpc on
```

With PHP running as CGI/phpsuexec manipulating the PHP settings is still possible however it can not be done with .htaccess. Using .htaccess with php\_value entries within it will cause a 500 internal server error when attempting to access the scripts. This is because php is no longer running as an apache module and apache will not handle those directives any longer.

All php values should be removed from your .htaccess files to avoid the 500 internal server error. Creating a php.ini file to manipulate the php settings will solve this issue.  
What is a php.ini file and how do I go about making one"

The php.ini file is a configuration file that the server looks at to see what options have been turned on, off or set to a number different from the defaults that we have set for the server. While the name may seem advanced to those unfamiliar with it, it's simply a text file with the name php.ini

To create a php.ini file, just open up a text editor, add in the lines you need and save the file. You can name the file whatever you wish when saving. Once done, upload the file to the directory where your script is located and then rename it to php.ini

For example you can turn off the php setting "magic\_quotes\_gpc" with this line in php.ini:

```
magic_quotes_gpc = no
```

Troubleshooting

HELP my php script doesn't work or I have an error message.

1. Check that the php script that you are attempting to execute has permissions of no more than 755 - 644 will work just fine normally, this is not something that will need to be changed in most cases.
2. Check that the directory permissions that the script resides within is set to a maximum of 755. This also includes directories that the script would need to have access to also.
3. Check that you do not have a .htaccess file with php\_values within it. They will cause a 500 Internal server error, when attempting to execute the script.

The php\_values will need to be removed from your .htaccess file and a php.ini put in its place, containing the php directives as explained above.