By MVied

Published: 2008-11-27 12:28

The Useful Uses Of Mod Rewrite

Author: Mike Ems

In this tutorial, I'm going to be teaching what mod rewrite is and few examples of its uses. Mod rewrite is a powerful tool and one of the simplest ways to make your website more SEO friendly.

What is Mod Rewrite?

Mod rewrite is a part of apache servers that can rewrite requested URLs on the fly. As it supports an endless number of rules that in turn have unlimited attached rule conditions it is very flexible and an important URL manipulation mechanism. It can be used for internet users and for search engine friendly URLs. This increases the chance of the database driven website to be indexed.

Enabling / Installing Mod Rewrite

The first thing we want to do is make sure that we have mod_rewrite on our server. By default, Apache has mod_rewrite installed, but not enabled. The easiest way to test if mod_rewrite is enabled on your server is to create a .htaccess file in a test directory such as `/modrewrite-test/.htaccess` with the following inside:

```
Options +FollowSymLinks
RewriteEngine On
```

Now attempt to browse to the subdirectory. One of two things could happen:

- No errors Congrats mod_rewrite engine is now enabled.
- 500, Internal Server Error If you get this message then mod_rewrite was not installed/enabled on your server.

If you found that mod_rewrite was not installed on your server, follow these steps.
- Find the `httpd.conf` file (usually you will find it in a folder called `conf`, config or something along those lines)
- Inside the `httpd.conf` file uncomment the line `LoadModule rewrite_module modules/mod_rewrite.so` (remove the pound '#' sign from in front of the line)
- Also find the line `ClearModuleList` is uncommented then find and make sure that the line `AddModule mod_rewrite.c` is not commented out.

After these steps, restart your apache server with the following command.

```
/etc/init.d/httpd restart
```

You should now have mod_rewrite successfully enabled. Be sure to test mod_rewrite again once apache server has been restarted.

Using Mod Rewrite To Always Remove / Add WWW

One easily fixed problem that many websites have is duplicate results in search engines. In any given search engine, `http://www.example.com/` and `http://example.com` are two completely different websites. This is a big problem because the content in your website is identified as duplicate content, and that brings your search engine rankings down. Using our new friend, mod_rewrite, we can assess this issue.

At the top of any .htaccess file, you need to turn Mod Rewrite on. This should always be at the top of your .htaccess file. You only need to turn Mod Rewrite on once.

```
RewriteEngine On
```

Additionally, you need to set your Rewrite Base. This is normally the directory in which the .htaccess file is located. If your .htaccess file is located in your root folder, your Rewrite Base would look like this:

```
RewriteBase /
```

Now then, there are two options to choose from. You can either choose to always remove the www, or always keep it. To always remove www (my preferred method), create a .htaccess file in your root directory. Inside of that file, add the following:

```
RewriteCond %{HTTP_HOST} ^www.example.com$ [NC]
RewriteRule ^(.*)$ http://example.com/$1 [R=301,L]
```

Likewise, if you would like to always add www, add the following to your .htaccess file:

```
RewriteCond %{HTTP_HOST} ^example.com$
RewriteRule (.*) http://www.example.com$1 [R=301]
```

Be sure to change `example.com` to your domain.

Using Mod Rewrite to Clean Up Your URL's

The most popular use of Mod Rewrite is using it to clean up your website's URL's. As you may or may not know, search engines don't particularly care for URL's with a lot of arguments like many websites serve. Example:

```
http://www.example.com/index.php?p=about
```

With the power of the mod_rewrite apache module, we can make this URL much cleaner.

```
http://www.example.com/about.html
```

To accomplish this, we get to see our good friend regular expressions. I know you're really excited now. To accomplish the rewrite above, I added the following to my .htaccess file after the add/remove www lines.

```
RewriteRule ^([^/]+).html$ /index.php?p=$1 [QSA,L,NC]
```

Let's take a closer look at this line so that we can really understand what it's doing.

```
([^/]+)
```

This is the regular expression to match anything except a forward slash. This is the easiest way to match anything that will fall between two forward slashes when dealing with rewriting URL's.

```
.html
```

This rewrites the url to have .html at the end. This gives the appearance of a regular .html file to your php file.

```
/index.php?p=$1
```

This is the part of the rewrite which is the originating location. Each time you have a regular expression in the first part matching a certain string, you should have it referenced in this part. The `$1` references the first regular expression in the first half of the line. So any value that is assigned to the variable p is placed where the first regular expression is in the first half of the line.

If you wanted your URL to look like a directory, you would use this line instead of the one given previously.

```
RewriteRule ^([^/]+)/?$ /index.php?p=$1 [QSA,L,NC]
```

This would return URL's like this:

`http://www.example.com/about/`

Adding a Trailing Slash To Your URL's

The main point of adding a trailing slash to your URL's is that it increases server performance. When a trailing slash is added to the end of a URL that does not end in a file extension, it tells the server to only make on call to itself, thus decreasing server traffic.

To always add a trailing slash to a URL that does not end in a file extention, such as `http://www.example.com`, add the following to your .htaccess file in your root directory.

```
rewriteCond $1 !/$
rewriteCond %{REQUEST_FILENAME}/ -d
rewriteRule (.+) http://www.example.com/$1/ [R=301,L]
```

The first line checks to see if there is no trailing slash. The second line checks to see if a directory exists with the given URL if the trailing slash is added. The third line adds the trailing slash and redirects the user.