Mod Rewrite Subdomains

You might have noticed some sites having their URLs written as if they were different subdomains of the same domain. For example, for a news site that has the dynamic links of the following form: http://yournewsdomain.com?topic=it - link that goes to IT news page

http://yournewsdomain.com?topic=business - link that goes to business news page

, to be rewritten in this form: http://it.yournewsdomain.com

http://business.yournewsdomain.com

You can achieve this without actually creating the subdomains on your server. This can be done with Apache mod_rewrite.

For this method to work you need a server that runs Apache and has mod_rewrite module installed. Also, if you don't have access to server configuration files, then you might need some help from your hosting support for the second step and possibly with the fists step of this tutorial.

Step 1. Create a wildcard DNS A record in your DNS Zone.

A wildcard DNS record is a record reprezented by the character "*" , that will match all non existing subdomain names. For example *.yournewsdomain.com will match for key1.yournewsdomain.com , key2.yournewsdomain.com and so on.

You will need to edit the DNS zone records for your domain and add an A record for *.yournewsdomain.com. , pointing to the IP for your domain.

If your domain is on a shared hosting account, then you will need to search in your control panel for a section for DNS management. Enter a new A type DNS record, set as name "*" and enter the IP for your domain. If you cannot find a DNS management section on your hosting account, then you will have to ask your hosting support to add it for you.

Step 2. Setup wildcard ServerAlias

You must now configure Apache server to serve the pages that come for any subdomain of your domain. Do this by adding a wildcard ServerAlias.

Add the following line to httpd.conf VirtualHosts section for your domain:ServerAlias www.yournewsdomain.com yournewsdomain.com *.yournewsdomain.com

Again, if you are on a shared hosting and don't have access to httpd.conf file, and no means in your control panel to edit your VirtualHosts section, then you must ask help from your hosting support.

Step 3. Add rewrite rules to your .htaccess file

First you must prepare your .htaccess file . If the file does not exist, create it.

You might already have these lines in your .htaccess files, if you do don't add them again. Options +FollowSymLinks

You need this option enabled for your rewrite rules to work. But in most cases is already enabled in the configuration files for the http server. RewriteEngine On

Use this line if RewriteEngine is not already enabled on your server. RewriteBase /

If your code is in the root of your domain, then use / as the rewrite base, like in the line above. If it is in a folder, then use

the path to that folder, for example:RewriteBase /subdomain/
Add the actual rewrite rules: RewriteCond %{HTTP_HOST} !www.yournewsdomain.com$ [NC]
RewriteCond %{HTTP_HOST} ^(www.)?([a-z0-9-]+).yournewsdomain.com [NC]
RewriteRule (.*) index.php?topic=%2 [NC,QSA]

First 2 lines add the condition for the actual rewrite rule (line 3) to come in place.

First line will add an exception for the subdomain www. Add here a similar line for each subdomain that you don't want to follow the same rule.

Second line adds the condition that the URL is in a form that looks like a subdomain , with or without the www string in front (http://anything.yournewsdomain.com).

The actual rule is the third line, and will translate any URL that matches the above conditions to a form: http://yournewsdomain.com?topic=%2.

Some things explained in the rewrite rules:

1. %2 is a back-reference that is created in the second condition. Back-references are created by groups of parentheses in the condition pattern. So in this case, the first parenthesis (www.) will create first back-reference that can be used with %1, and the second one ([a-z0-9-]) , that matches the subdomain like form, will create the second back-reference, which we use with %2.

2. The flag NC at the end of the conditions and rule means that the comparisons should be made case insensitive (No Case).

3. The flag QSA at the end of the rule means that if there are additional queries at the end of URL string, they should be added to the URL when rewritten (Query String Append).

You might need to make adjustments to the rewrite rule form, for example instead of index.php you have other landing page, or instead of the parameter "topic" you have other.

Stay tuned for more mod_rewrite tutorials soon!