URL redirection is a common and unfortunate necessity of running web sites. Sometimes sites go through renovation, and sometimes the entire infrastructure needs to be changed. Perhaps you're moving to more SEO-friendly URL's, or perhaps you just need to shuffle some files around for organizational reasons. Now if someone has an interior page of your site bookmarked, that person will get a "Page Not Found" error (also called a 404 error). New visitors can also become confused if they've reached your site from a deep link on another site, or from a search engine which now has an outdated index of your site. And that raises another problem - Google doesn't like it.

The Solution: 301 Redirect

Create a 301 redirect. This will allow you to redirect all of your old file locations to their new, proper locations. When the Google spider comes along, it'll also follow your 301 redirects, and this will let Google know that your pages have been moved, and the Google index will be updated with the new file locations.

301 Redirect Using .htaccess

The simplest way to do this is using an .htaccess file. This is a simple text file (named .htaccess) that the server reads every time it loads a page inside the directory (and subdirecties) in which the .htaccess file is placed. Basically, when the server loads a page, it first reads the server configuration file, then it looks for any .htaccess files to override the settings in the configuration file. This allows you to change server settings only within the directory you've placed the .htaccess file. If you have an .htaccess file in your root directory (let's call it public_html - it usually is), the settings in that file will override the server configuration. If the page being loaded is in a subdirectory, and that subdirectory contains an .htaccess file, then those settings will override both the server configuration and the root folder's .htaccess file.

If your root/public_html directory already contains an .htaccess file, then you'll simply open that file with a text or html editor (such as notepad or Dreamweaver) and add any necessary lines to that file. To create a new .htaccess file, just create it the same way you would create a text file. To simplify filename associations, I tend to create the file with the name: htaccess.txt. Then I can double click it to open it, edit it, and after I save it I then rename the file to ".htaccess".

301 Redirect: yoursite.com to www.yoursite.com

This prevents search engine confusion (which can potentially result in duplicate content

penalties) as it indexes both http://yourdomain.com and http://www.yourdomain.com. This method will redirect all traffic coming in without the www to the same URL with the www.

```
RewriteEngine On
 RewriteBase /
 RewriteCond %{HTTP_HOST} !^www.xyz-site.com$ [NC]
 RewriteRule ^(.*)$ http://www.xyz-site.com/$1 [L,R=301]
```

301 Redirect: www.yoursite.com to yoursite.com

This serves the same purpose as the last example, but instead redirects any traffic with www to the same url without the www.

```
RewriteEngine On
 RewriteBase /
 RewriteCond %{HTTP_HOST} !^xyz-site.com$ [NC]
 RewriteRule ^(.*)$ http://xyz-site.com/$1 [L,R=301]
```

301 Redirect: Redirecting individual pages

This method will redirect a single page to its new location.

```
Redirect 301 /oldpage.html http://www.yoursite.com/newpage.html
```

If you would like to redirect more individual pages, merely add more lines:

```
Redirect 301 /oldpage1.html http://www.yoursite.com/newpage1.html
 Redirect 301 /oldpage2.html http://www.yoursite.com/newpage2.html
 Redirect 301 /oldpage3.html http://www.yoursite.com/newpage3.html
 Redirect 301 /oldpage4.html http://www.yoursite.com/newpage4.html
```

Redirecting with PHP

This method works by putting a piece of code at the top of your old page. This means that the old page must still exist on the server, and inside that page is the following piece of code, which will create a 301 redirect to the new file location. This code should be the first text on the page.

```
<?php
 Header( "HTTP/1.1 301 Moved Permanently" );
 Header( "Location: http://www.yournewsite.com" );
```

```
exit(0);
?>
```

Redirecting with ASP.NET

This works the same as the PHP method, but on a Windows server running ASP.NET.

```
<script>
private void Page_Load(object sender, System.EventArgs e)
{
Response.Status = "301 Moved Permanently";
Response.AddHeader("Location","http://www.yournewsite.com");
}
</script>
```

Redirecting with ASP

This works the same as the PHP method, but on a Windows server running ASP.

```
<@ Language=VBScript %>
<%
Response.Status="301 Moved Permanently" Response.AddHeader "Location",
" http://www.yournewsite.com"
%>
```

Redirecting with ColdFusion

This works the same as the PHP method, but on a server running ColdFusion.

```
<cfheader statuscode="301" statustext="Moved permanently">
<cfheader name="Location" value="http://www.yournewsite.com/">
```

Redirecting with Perl

This works the same as the PHP method, but uses Perl.

```
$q = new CGI;
print $q->redirect(" http://www.yournewsite.com/ ");
```

Other Redirection Methods

I would strongly advise against using redirection methods such as Javascript and meta refresh. These methods have been abused and the search engines frown on them. A 301 redirect is the proper way to redirect a page, and the techniques described above should allow you to do this on any server. There's really no reason to ever use a meta refresh or javascript redirect.