# Editar htaccess para crear direcciones URL amigables


## Autores del manual

Este manual ha sido realizado por los siguientes colaboradores de DesarrolloWeb.com:

**Miguel Angel Alvarez**
Director de DesarrolloWeb.com
http://www.desarrolloweb.com
(8 capítulos)

# *Introducción a .htaccess y a las URLs amigables a buscadores*

Algunas URL son mejor consideradas por los motores de búsqueda tipo Google que otras. Esto ya lo habíamos comentado en nuestro manual de promoción web. Por ejemplo, URLs como estas no son muy atractivas para la promoción de las páginas:

www.dominio.com/articulos/muestra.php?id=23
www.dominio.com/pueblos/pueblo.php?nombre=torrelodones

Aunque una URL con parámetros tenga contenidos específicos, los buscadores no las puntúan tan bien como si fueran los mismos contenidos con URLs sin paso de parámetros.

www.dominio.com/articulos/23.php
www.dominio.com/pueblos/torrelodones.php

Incluso, la dirección del artículo 23.php, podría ser mejor si incluyese en la propia URL alguna palabra clave, por ejemplo, si el artículo hablase sobre coches antiguos, una dirección mejor sería:

www.dominio.com/articulos/coches_antiguos.php

> **Referencia:** en nuestro manual de posicionamiento en buscadores explicamos con detalle las ventajas de utilizar URLs amistosas a buscadores.

Al ver una URL como esta, parece como si en el servidor web, en el directorio articulos tuviéramos un documento llamado coches_antiguos.php. En principio tendría que ser así para que esta dirección fuese correcta y no arrojase un error 404 de página no encontrada. Ahora imaginemos que tenemos cientos o miles de artículos. Entonces deberíamos que tener dentro del directorio articulos sus correspondientes cientos o miles de archivos. Esto no es muy óptimo de cara al mantenimiento de la página, porque cada vez que se publica un artículo se debería crear el correspondiente archivo en el servidor, en el directorio adecuado.

Puede que ya sepamos acerca de esta idea de las URLs amistosas para buscadores, pero ahora vamos a comentar sobre cómo realizarlas ayudándonos del archivo .htaccess y la instrucción RewriteRule. Con ello podemos crear URLs fácilmente en el servidor sin que se correspondan con documentos que tengamos en la propia estructura de directorios. Es decir, Apache será capaz de servir URLs que realmente no existen en el servidor, haciendo la redirección a otros archivos y procesando mediante PHP, u otro lenguaje de programación compatible, para mostrar unos contenidos u otros dependiendo de la URL que se está intentando acceder.

La inclusión de palabras clave dentro de las URLs y conseguir evitar el paso de parámetros, todo mediante el htaccess, es una técnica cada vez más usada, que debemos conocer y utilizar para que nuestra página sea fácilmente promocionable en buscadores.

El .htaccess es un archivo de texto propio de Apache, que se coloca en cualquiera de los directorios de publicación del servidor web y afecta al directorio donde esté colocado y a todos sus subdirectorios. Con .htaccess se pueden configurar muchos temas variados para esos directorios, sin tener que tocar el archivo httpd.conf de Apache, que es donde se guardan las configuraciones generales del servidor.

Nosotros vamos a explicar el uso de la instrucción RewriteRule, aunque htaccess sirve para configurar otros muchos asuntos.

Se puede obtener más información sobre .htaccess en DesarrolloWeb.com:

- El fichero httpd.conf
- Htaccess y páginas dinámicas

El manual continúa con explicaciones prácticas y detalladas sobre el trabajo con htaccess.

*Artículo por **Miguel Angel Alvarez***

## *Redirigir URLs a buscadores hacia URLs con paso de parámetros con RewriteRule*

Ahora vamos a ver cómo realizar con htaccess unas reglas de redirección, de las URLs pensadas para que estén bien promocionadas en buscadores hacia direcciones que nos sean fáciles de procesar y mantener con PHP.

> **Nota:** Hablamos de PHP porque estos trucos con htaccess son para el servidor Apache y el lenguaje de programación típico de este servidor web es PHP.

El archivo htaccess es un fichero de texto que se llama .htaccess, es decir, su nombre comienza con un punto. Se puede crear con cualquier editor de textos y se coloca en el directorio donde deseamos que afecten las configuraciones indicadas dentro del fichero. Si lo colocamos en el directorio de publicación raíz del dominio, afectará a todos los directorios del dominio, pues este archivo modifica el directorio que lo contiene y todos los subdirectorios.

El mecanismo para crear unas redirecciones pensadas para un mejor posicionamiento en buscadores es el siguiente. En el archivo .htaccess se define una regla de redirección interna, con esta sintaxis:

RewriteRule url_patron url_destino

RewriteRule es la instrucción para definir una redirección, que recibe dos parámetros: url_patron y url_destino. En el primero se especifica una URL patrón. Cuando la dirección a la que se está accediendo cumple el patrón indicado en url_patron, se redirecciona internamente hacia la URL de destino url_destino. La página se procesa en el archivo indicado en url_destino.

El patrón es una expresión regular que corresponde con un conjunto de URLs posibles. Cuando el visitante intenta acceder a una URL que cumple esa expresión regular, internamente se procesa la página a través de la url_destino. Decimos que es un proceso interno, porque nadie se entera que la URL se está procesando a través de otro archivo, pues la dirección que aparece en el navegador siempre será la original y los buscadores no podrán detectar que Apache ha realizado una redirección interna para procesar la página.

Así pues, para los visitantes y los buscadores que indexen los contenidos de nuestra web, las direcciones que están accediendo son las amigables para buscadores, a pesar que en realidad

esos documentos no existan dentro de la estructura de directorios del servidor y el procesamiento real de la página se realice en otra URL de destino.

Un ejemplo de instrucción RewriteRule podría ser el siguiente:

RewriteRule ^articulos/(.+)\.php codigo/ver_articulo.php?nombre=$1

Como se ha dicho, la primera parte es el patrón que debe cumplirse para que se realice la redirección interna. Dicho patrón es una expresión regular que tiene una sintaxis especial.

> **Nota:** La parte más complicada del trabajo con RewriteRule de htaccess es justamente la creación de las expresiones regulares. Nosotros no vamos a explicar en este artículo la creación de expresiones regulares, pues resulta bastante compleja, pero realizaremos suficientes ejemplos como para conocer los casos más típicos. No obstante, existen referencias sobre expresiones regulares que podemos visitar para obtener más información. Visitar la categoría de expresiones regulares en nuestro directorio.

La dirección que se está accediendo en el servidor (URL) tiene que concordar con la expresión regular para que se realice la redirección. En la primera parte de la instrucción RewriteRule del ejemplo indicado anteriormente teníamos esta expresión regular:

^articulos/(.+)\.php

Para explicar esta expresión regular creo que es mejor verla por partes.

La primera parte sería "^articulos/". Esto quiere decir todas las URL que comiencen por "articulo/" (la palabra artículo seguida de una barra). El carácter ^ significa el comienzo de una expresión.

La segunda parte es (+.). El "+" significa una o más repeticiones de algo. El "." significa cualquier carácter, luego "+." significaría uno o más repeticiones de cualquier carácter, es decir, cualquier conjunto de uno o más caracteres. El paréntesis sirve para agrupar expresiones. Luego veremos para qué nos puede servir esa agrupación.

Por último, en la expresión regular tenemos "\.php". La contrabarra es un carácter de escape que sirve para que el "." siguiente no sea considerado como cualquier carácter, sino como un punto sin más. Entonces, esto quiere decir ".php".

En conjunto la expresión regular significa cualquier cosa que empiece por "articulos/", seguido de cualquier carácter o conjunto de caracteres, seguido de ".php".

Con esta expresión regular concordarían URLs como estas:

articulos/loquesea.php
articulos/otra-cosa.php
articulos/1234-xx_zz.php

Decíamos que la parte con el (+.) quiere decir cualquier carácter o conjunto de caracteres. Además, con este "comodin" se puede construir la url_destino. Fijémonos en la segunda parte del RewriteRule:

codigo/ver_articulo.php?nombre=$1

Simplemente se ha indicado otra dirección donde hay un archivo PHP que se va a encargar de

procesar todas las URLs que cumplan el patrón explicado anteriormente. Si nos fijamos, dentro de la url_destino tenemos un $1. Esta variable se sustituye por lo que había dentro del (+.) indicado en el patrón de la expresión regular.

Por ejemplo, una url de esta forma:

www.dominio.com/articulos/plantas-medicinales.php

Se procesará en la dirección:

www.dominio.com/codigo/ver_articulo.php?id=plantas-medicinales

Si nos fijamos, la URL de destino realiza el paso de parámetros, para que todos los artículos se procesen en el mismo archivo, indicando en la dirección algo que sirva para identificar inequívocamente el artículo que se desea ver. Sin embargo, ese paso de parámetros se realizará internamente y de manera transparente para el usuario que estará teniendo la impresión que la URL que está procesando es la original que ha escrito en la barra de direcciones del navegador.

Como se ha visto, el $1, extraído de la URL original en la parte del patrón de la expresión regular que se corresponde con el (+.), se utiliza para construir la URL que se va a encargar de procesar la página.
**Warning**: mysql_num_rows(): supplied argument is not a valid MySQL result resource in **/home/chs/desarrolloweb.com/home/librerias/imprimir_manual_completo2.php** on line **148**

## *Ejemplo más complejo de RewriteRule*

En el anterior capítulo vimos como trasladar con RewriteRule parte de una URL estática a una URL con paso de parámetros. Para ello definíamos en el patrón una agrupación con (+.) y luego la utilizábamos con $1 en la URL redirigida con paso de parámetros. Ver el anterior artículo para más información.

En algunas ocasiones no basta con enviar un parámetro para que la página reciba todos los datos que necesita para mostrar los contenidos específicos.
Si en la expresión regular tuviéramos otros (+.) entonces en la url_destino tendríamos que utilizar $1 para el primer (+.) que haya en la expresión regular y $2 para el segundo, $3 para el tercero y así sucesivamente.

Por ejemplo, aquí podemos ver una sentencia RewriteRule que utiliza dos partes con (+.) para hacer el patrón.

RewriteRule ^agenda/(.+)/(.+)\.html codigo/cita.php?mes=$2&anio=$1

Esta expresión regular quiere decir: Cualquier URL que comience por "agenda/", seguida de cualquier cosa, luego una barra y cualquier otra cosa, acabado en ".html" (fijarse la contrabarra antes de ".html" que es un carácter de escape para indicar que el "." de antes de "html" es un punto, en lugar del codigo especial que suele significar cualquier caracter. Ver el artículo anterior para más explicaciones sobre las expresiones regulares de este estilo.

Como se puede ver, las direcciones estáticas amigables a buscadores no tienen por que finalizar en .php. En este caso finaliza en .html, pero podría acabar de cualquier otra forma, como .htm, .php3, .shtml o en un directorio.

En el caso del ejemplo anterior, una dirección con esta forma:

www.dominio.com/agenda/2006/7

Se hará corresponder con una url_destino así:

www.dominio.com/codigo/cita.php?mes=7&anio=2006

Aunque también podrían haber concordado otros modelos de direcciones como:

www.dominio.com/agenda/05/01
www.dominio.com/agenda/2006/marzo

Será nuestra responsabilidad hacer el tratamiento para que sólo las URL que queremos muestren la página correcta. Veremos en el proximo capítulo cómo hacer este tratamiento de posibles errores y cómo resolver otros posibles problemas del uso del htaccess.

*Artículo por **Miguel Angel Alvarez***

## No tener dos URL con los mismos contenidos

Uno de los problemas que podemos encontrar derivados del uso de htaccess para crear URLs amistosas a buscadores es que se de el caso de que distintas URLs muestren exactamente las mismas informaciones.

Por lo visto, tener en un dominio dos páginas con los mismos contenidos no está bien visto por los motores de búsqueda, que pueden considerarlo una trampa encaminada a confundirle y hacerle pensar que el sitio es más grande de lo que realmente es.

¿Cómo puede ser que dos páginas tengan los mismos contenidos?

Por ejemplo, pensemos en una regla como esta:

RewriteRule ^articulos/(.+)\.php codigo/ver_articulo.php?nombre=$1

Esto quiere decir que existirán URLs como estas:

www.dominio.com/articulos/plantas-decorativas.php
www.dominio.com/articulos/plantas-medicinales.php
www.dominio.com/articulos/plantas-aromaticas.php
…

Imaginemos que alguna persona se equivoca y nos pone un enlace a un artículo que no existe:

www.dominio.com/articulos/plantas-decor.php
www.dominio.com/articulos/plantas-medicina.php

Nosotros en la página que muestra los artículos "ver_articulo.php" tendremos que comprobar si existe un artículo con ese nombre.

```
//tengo que ver si este articulo tiene nombre
$ssql = "select * from articulo where nombre = $nombre ";
$rs = mysql_query($ssql);
if (mysql_num_rows($rs)==0){
//es que no existe un articulo con ese nombre
echo "Error. No tenemos ese artículo";
}
```

Con este código, cada vez que se escriba una URL con un error nos mostraría los mismos contenidos: el mensaje "Error. No tenemos ese artículo". Esto es algo que pretendíamos evitar.

Para evitar mostrar en URLs que no existen realmente los mismos contenidos, una solución es hacer una redirección a una página de error.

```
if (mysql_num_rows($rs)==0){
//es que no existe un script con ese id
header ("location: /error_articulo.php");
exit();
}
```

Así, cuando alguien se equivoque al componer la URL se mostrará un mensaje de error, pero para el buscador todos los mensajes se ofrecen desde la misma página web.

También podemos hacer que PHP envíe al navegador del usuario una cabecera con el error de página no encontrada (error 404 del HTTP).

```
header("HTTP/1.0 404 Not Found");
```

Son dos posibilidades, cada cual puede utilizar la que desee, aunque es posible que la del error 404 sea más útil y elegante.

*Artículo por **Miguel Angel Alvarez***

## *Redirecciones cíclicas con .htaccess*

Nuestro manual de crear URLs amigables a buscadores con htaccess continua comentando algunos de los errores típicos que se pueden cometer a la hora de definir las reglas de redirección.

En mi experiencia con .htacces he encontrado que es fácil hacer un redireccionamiento cíclico. Es decir, definir un RewriteRule con un patrón que redirecciona a una URL que sigue cumpliendo el patrón. Como esa segunda URL concuerda todavía con el patrón, se volverá a redirigir y seguirá cumpliendo el patrón, con lo que se volverá a redirigir. Y así indefinidamente.

Al escribir una dirección del dominio que cumpla el patrón, el resultado de este problema de redirección cíclica será que Apache nos presentará un error "Internal Server Error", pero no será mucho más descriptivo, por lo que podría complicarse identificar el problema. Fijémonos en esta regla de redirección:

```
RewriteRule ^dir/(.+)\.php dir/codigo.php?articulo=$1
```

Quiere decir que cualquier URL del directorio "dir" que acabe en .php, se redirigirá a dir/codigo.php?articulo=xxx. Por ejemplo:

http://www.midominio.com/dir/loquesea.php

Se redirigirá a: http://www.midominio.com/dir/codigo.php?articulo=loquesea

Como se puede ver, la URL a la que se redirigirá cumple también el patrón, porque la URL a la que se accede también está dentro del directorio "dir" y acaba en PHP. (No se tiene en cuenta para comprobar el patrón el parámetro que se pasa por la URL)

Ese RewriteRule sería incorrecto, por padecer una redirección cíclica. Para solucionarlo podemos hacer varias cosas. Por ejemplo, redirigir a una URL que no esté dentro del directorio dir.

```
RewriteRule ^dir/(.+)\.php codigo-dir/codigo.php?articulo=$1
```

Esto funcionaría bien. La URL de antes http://www.midominio.com/dir/loquesea.php

Será redirigida hacia http://www.midominio.com/codigo-dir/codigo.php?articulo=loquesea

Otra solución que a veces he utilizado es redirigir direcciones acabadas en .html a direcciones acabadas en .php. Como las terminaciones son distintas, no habrá redirecciones cíclicas:

```
RewriteRule ^dir/(.+)\.html dir/codigo.php?articulo=$1
```

Por ejemplo, para la URL: http://www.midominio.com/dir/otracosa.html

La redirigirá a: http://www.midominio.com/dir/codigo.php?articulo=loquesea

Esta última URL, como no acaba en .html no cumplirá el patrón y no tendremos la redirección cíclica.

*Artículo por **Miguel Angel Alvarez***

## Cambio de URL redirección 301 con .htaccess

A veces nos vemos en la necesidad de cambiar las direcciones de nuestras páginas, porque hayamos realizado una reestructuración del sitio, una reprogramación o un cambio de dominio. No cabe duda que esto siempre es un engorro, lo mejor sería conservar las direcciones antiguas, que las conocen nuestros usuarios y están correctamente posicionadas en

buscadores, pero en ocasiones no tenemos más remedio.

En esos casos, lo más adecuado es facilitar tanto a usuarios como motores de búsqueda la localización de las nuevas URLs que sustituyen a las viejas. Existen varias maneras de hacer esto, unas más adecuadas que otras, pero parece ser que, de cara a buscadores como Google, lo más adecuado es realizar una redirección 301 "moved permanently". Así les estamos informando que los contenidos han cambiado de localización permanentemente, de modo que los motores de búsqueda actualizarán las direcciones en sus bases de datos. De cara a los usuarios, con una redirección 301 sus navegadores cambiarán las URL por las nuevas de manera transparente para los usuarios, es decir, sin que tengan que hacer nada.

En este artículo veremos cómo realizar una redirección 301 ayudándonos del archivo htaccess de Apache.

> **Referencia:** Tenemos algunas informaciones sobre .htaccess en DesarrolloWeb.com.
>
> Editar htaccess para crear direcciones amigables
> http://www.desarrolloweb.com/manuales/htaccess-para-urls-amigables.html
>
> Manual de Apache
> http://www.desarrolloweb.com/manuales/41/

Realizar una redirección 301 con htaccess es muy simple. Se puede hacer con una línea como esta en el archivo:

```
redirect 301 /url_antigua.html http://www.dominio-nuevo.com/url-nueva/
```

Como se ha visto, se lanza un comando redirect, tipo 301 y luego se indican tanto la url antigua como la nueva. La antigua simplemente se indica con la URL relativa al archivo .htaccess. La nueva URL se indica de manera absoluta, comenzando con http://.

También podemos hacer redirecciones en masa más complejas utilizando patrones. Si los patrones concuerdan, entonces se realiza la redirección.

Por ejemplo, si queremos que cualquier URL de un dominio se redirija a la portada o página raíz de otro dominio, podremos hacer esto:

```
redirectMatch 301 ^(.*)$ http://www.desarrolloweb.com
```

Esto, en un dominio llamado por ejemplo domantiguio.com, redireccionaría cualquier URL como http://domantiguo.com/loquesea/ a la URL http://www.desarrolloweb.com. O bien una URL como http://domantiguo.com/dir/otracosa.html se redirigiría a http://www.desarrolloweb.com.

Ahora, si deseamos hacer una redirección de una URL de un dominio a la misma URL, pero en otro dominio, podríamos hacer algo como esto:

```
redirectMatch 301 ^(.*)$ http://www.desarrolloweb.com$1
```

Esto redirigiría algo como http://domantiguo.com/loquesea/ a la URL http://www.desarrolloweb.com/loquesea. Una URL como http://domantiguo.com/dir/otracosa.html la redirigiría a http://www.desarrolloweb.com/dir/otracosa.html.

*Artículo por **Miguel Angel Alvarez***

## *Carácter $ para finalizar los patrones de redirección con .htaccess*

Siguiendo con los comentarios y consejos del manual de .htaccess, hay que comentar el uso del carácter $ en los patrones, para especificar el final de una URL.

Para explicarlo mejor vamos a ver un ejemplo.

**Veamos esta regla de .htaccess**

RewriteRule ^articulos/(.+)\.html codigo/muestra_articulo_htaccess.php?id=$1

Esta regla dice que todas las URL como estas:

www.dominio.com/articulos/x.html
www.dominio.com/articulos/yyy.html

Serán procesadas con el archivo:

www.dominio.com/codigo/muestra_articulo_htaccess.php?id=x
www.dominio.com/codigo/muestra_articulo_htaccess.php?id=y

Pero el tema es que este patrón de htaccess también casa con otras url como estas:

www.dominio.com/articulos/x.htmlkk
www.dominio.com/articulos/x.html-loquesea

Esto es porque el patrón coincide, porque corresponde con la regla. Es decir la regla dice que empieza la URL de después del dominio por "articulos/" + cualquier conjunto de caracteres + ".html".

Tendríamos que especificar de alguna forma que la URL debe terminar en ".html" y nada más y para eso podemos utilizar "$" al final de la regla.

Ahora esta otra regla .htaccess más completa

RewriteRule ^articulos/(.+)\.html$ codigo/muestra_articulo_htaccess.php?id=$1

En este caso hemos indicado con el "$" después de \.html que la URL debe finalizar ahí.

**Algo como esto sí que concordaría la regla htaccess:**

www.dominio.com/articulos/algo.html

Pero esto no se procesaría por el .htaccess, porque no acaba en .html:

www.dominio.com/articulos/algo.htmlkk

*Artículo por **Miguel Angel Alvarez***

## *Patrones de expresiones regulares para htaccess*

Las expresiones regulares son, digamos, un hueso duro de roer. Utilizan un lenguaje complicado que a veces se hace difícil de entender o de especificar. Por ello, es habitual que el desarrollador tenga ciertos problemas a la hora de crear sus patrones de expresiones regulares para utilizar en el .htaccess. A mi mismo me resultaba muchas veces complicado escribir las reglas, hasta que entendí un poco el lenguaje de expresiones regulares y creé una serie de reglas de uso común, que luego suelo reutilizar en mis archivos .htaccess.

En este artículo pienso proporcionar una serie de ejemplos de reglas htaccess, o de patrones de expresiones regulares que podemos utilizar en nuestras páginas web. No obstante, cabe recordar para los lectores que tenemos diversos recursos interesantes para documentarse sobre estos asuntos en:

- Manual de htaccess
- Categoría de expresiones regulares, con diversos manuales y artículos sobre el tema.

Sin más, empecemos dando los distintos ejemplos:

### Ejemplo 1 de expresión regular para htaccess

```
RewriteRule ^resultados/pagina_pg([0-9]+)\.php resultados/index.php?&_pagi_pg=$1
```

Esta regla puede servir bien para paginadores, porque crearmos URLs amigables que tienen el número de la página de resultados que se desea ver. Esta regla redirige todo lo que sea como:

www.midominio.com/resultados/pagina_pg5.php
www.midominio.com/resultados/pagina_pg23.php
www.midominio.com/resultados/pagina_pg19992.php
… y cualquier otra combinación de números, con tantos dígitos como sea necesario.

A direcciones como estas:

www.midominio.com/resultados/index.php?&_pagi_pg=5
www.midominio.com/resultados/index.php?&_pagi_pg=23
www.midominio.com/resultados/index.php?&_pagi_pg=19992

### Ejemplo 2 de expresión regular para htaccess

```
RewriteRule ^platos/letra_([a-z])\.php$ codigo/platos_inicial.php?letra=$1
```

Esta regla htaccess es parecida a la anterior, sólo que en vez de números gestiona iniciales y además, como no tiene el signo "+", sólo acepta una letra.

Acepta URLs como estas:
www.midominio.com/platos/letra_a.php

www.midominio.com/platos/letra_x.php

Y redirige estas URLs a direcciones como estas:

www.midominio.com/codigo/platos_inicial.php?letra=a
www.midominio.com/codigo/platos_inicial.php?letra=x

Pero no acepta más de una letra en la inicial y sólo acepta minúsculas. Osea, estas direcciones no concordarían con el patrón de expresión regular:

www.midominio.com/platos/letra_xy.php
www.midominio.com/platos/letra_A.php

**Ejemplo 3 de expresión regular para htaccess**

```
RewriteRule ^platos/pais_([a-z_-]+)\.php$ codigo/platos_pais.php?nombre_pais=$1
```

Esta regla avanza un poco en la regla anterior, porque permite cualquier número de caracteres y además también acepta que se coloquen guiones bajos y medios.

Acepta URLs como estas:

www.midominio.com/platos/pais_a.php
www.midominio.com/platos/pais_abc.php
www.midominio.com/platos/pais_a-b_c.php

Estas direcciones las redireccionaría internamente a estas URL:

www.midominio.com/codigo/platos_pais.php?nombre_pais=a
www.midominio.com/codigo/platos_pais.php?nombre_pais=abc
www.midominio.com/codigo/platos_pais.php?nombre_pais=a-b_c

Esta regla no aceptaría nombres de países que tuvieran una mayúscula.

*Artículo por **Miguel Angel Alvarez***

## *Más ejemplos de reglas .htaccess*

En el artículo anterior estuvimos dando una serie de ejemplos para la creación de expresiones regulares que podemos utilizar en archivos .htaccess. Vimos una serie de patrones simples y ahora vamos a seguir mostrando otros patrones, también sencillos, pero un poco más elaborados.

Antes que nada, deberíamos leer el artículo anterior, si es que no lo hemos hecho ya: Patrones de expresiones regulares para htaccess.

**Ejemplo 4 de regla htaccess**

```
RewriteRule ^platos/pais_([a-zA-Z_-]+)\.php$ codigo/platos_pais.php?nombre_pais=$1
```

Esta regla es similar a la anterior, pero ahora sí que estamos aceptando que algunos caracteres del nombre del país vengan en mayúsculas. Por ejemplo, acepta estas URLs:

www.midominio.com/platos/pais_Espana.php
www.midominio.com/platos/pais_Reino-Unido.php

Que redirigiría a URLs como estas:

www.midominio.com/codigo/platos_pais.php?nombre_pais=Espana
www.midominio.com/codigo/platos_pais.php?nombre_pais=Reino-Unido

**Ejemplo 5 de regla htaccess**

```
RewriteRule ^platos/nombre_([a-zA-Z0-9_-]+)\.php$ codigo/platos_nombre.php?nombre=$1
```

Esta regla .htaccess complica todavía un poco más la regla anterior, con la posibilidad que también se pongan números en la expresión.

Redigirirá URLs como estas:

www.midominio.com/platos/nombre_paella.php
www.midominio.com/platos/nombre_PAELLA_2.php
www.midominio.com/platos/nombre_paella-mariscos.php
www.midominio.com/platos/nombre_Arroz3Delicias.php

Que enviará internamente el procesamiento de esas páginas a estas direcciones:

www.midominio.com/codigo/platos_nombre.php?nombre=paella
www.midominio.com/codigo/platos_nombre.php?nombre=PAELLA_2
www.midominio.com/codigo/platos_nombre.php?nombre=paella-mariscos
www.midominio.com/codigo/platos_nombre.php?nombre=Arroz3Delicias

**Ejemplo 6 de regla htaccess**

Este esquema visto en el anterior ejemplo lo podemos repetir las veces que queramos en URLs que tengan una serie de directorios, separados por barras. Será algo como esto:

```
RewriteRule ^([a-zA-Z0-9_-]+)/([a-zA-Z0-9_-]+)/([a-zA-Z0-9_-]+)\.html$ codigo/mascodigo/procesamiento.php?parametro1=$1&parametro2=$2&parametro3=$3
```

Esto acepta direcciones que tengan tres directorios distintos, uno detrás de otro, separado claro está por las barras. Los nombres de los directorios aceptan números, letras mayúsculas y minúsculas y guiones medios y bajos.

Acepta direcciones tan variadas como estas:

www.midominio.com/1/2/3.html
www.midominio.com/loquesea/OTRACOSA/122.html
www.midominio.com/a_b_c_1/A-B-C-2/A-b_C_987.html

Internamente, htaccess procesará esas reglas y redirigirá a url como estas:

www.midominio.com/codigo/mascodigo/procesamiento.php?
parametro1=1&parametro2=2&parametro3=3
www.midominio.com/codigo/mascodigo/procesamiento.php?
parametro1=loquesea&parametro2=OTRACOSA&parametro3=122
www.midominio.com/codigo/mascodigo/procesamiento.php?
parametro1=a_b_c_1&parametro2=A-B-C-2&parametro3=A-b_C_987

*Artículo por **Miguel Angel Alvarez***