http://www.netandsoftware.es/articulos-blog/articulos-del-blog/art-para-webmasters-y-joomla/505-htaccess-en-profundidad-la-guia-completa

0 - Indice

- Introducción al .htaccess
- Bloqueando y permitiendo acceso
- Re-escritura y Redirección
- Compresión y caché
- Otras funcionalidades del .htaccess
- Recomendaciones sobre el fichero .htaccess
- Simbología del fichero .htaccess

1.- Introducción

Hay muchísimos artículos en la red sobre el fichero .htaccess, así que es una buena forma de agrupar y reunir lo que realmente es interesante sobre este tema, mediante ejemplos claros que sirvan como ejemplo.

Primero, tenemos que decir que el fichero .htaccess es una verdadera 'navaja suiza' para el control del acceso a tu sitio web. Con ella se puede redireccionar a una persona, denegar el acceso, comprimir los ficheros, establecer una caché para los archivos, etc.

.htaccess significa acceso de hipertexto -hipertext access- y es un archivo de configuración de Apache. Apache es el software encargado de servir páginas web para plataformas Unix, Microsoft Windows, Mac, etc.

Advertencia: un error en el .htaccess puede hacer que no se vea la web hasta que se arregle dicho error.

Es evidente, pero importante, saber que en todos los ejemplos habrá que sustituir los nombres genéricos y las IP por las correctas en tu servidor. En muchos casos los dominios son del tipo dominio.com, ejemplo1.com, ejemplo2.com y similares, y las IP suelen ser 100.100.100.100, 100.101.102.103, y parecidas.

2.- Bloqueando y Permitiendo Acceso

Denegar el acceso a un directorio del servidor

Crear un fichero .htaccess dentro de la carpeta con las instrucciones:

```
#denegar todo acceso
deny from all
```

Permitir el acceso sólo a una IP

```
#denegar todo acceso excepto una IP
deny from all
allow from 100.100.100.100
```

Permitir el acceso sólo a un rango específico de IPs (forzado mediante la máscara de red)

```
#denegar todo acceso excepto a un rango de IP
deny from all
allow from 100.100.100.100/24
```

Bloquear el acceso a un archivo específico

```
#bloquear un fichero concreto
<Files archivo_a_bloquear.html>
  order allow,deny
  deny from all
```

Redireccionar visitantes a una dirección alternativa, menos a una IP específica

```
#Redireccionar a todos a otrositio.com excepto una IP concreta
ErrorDocument 403 http://www.otrositio.com
order deny,allow
deny from all
allow from 100.101.102.103
```

Denegar un rango completo de IP

Esto quiere decir que elimina 256 direcciones en el primer caso, 65536 en el segundo caso, y más de 16 millones de direcciones IP en el tercer caso

```
# Primer caso: deniega el acceso a 256 IPs, desde la 100.100.100.0 hasta la
100.100.100.255
order allow,deny
deny from 100.100.100.
```

```
# Segundo caso: deniega 65536 IP
order allow,deny
deny from 100.100.
```

```
# Tercer caso: deniega todas las IP que empiezan por 100, que son más de 16
millones
order allow,deny
deny from 100.
```

Permitir un rango concreto de IP a la web

```
# permitir acceso a rango 100.100.100.0 hasta 100.100.100.255
order deny,allow
allow from 100.100.100.
```

Denegar acceso de un dominio concreto

```
# bloquear acceso a un dominio
order allow,deny
allow from all
deny from .*dominio\.com.*
```

Denegar acceso a visitantes que vengan de un dominio concreto

```
# bloquear acceso visitas desde ejemplo1.com y ejemplo2.com
RewriteCond %{HTTP_REFERER} ejemplo1\.com [NC,OR]
RewriteCond %{HTTP_REFERER} ejemplo2\.com [NC,OR]
RewriteRule .* - [F]
```

Denegar acceso a una hora específica

```
# bloquear acceso una hora
RewriteCond %{TIME_HOUR} ^12$
RewriteRule ^.*$ - [F,L]
```

Denegar acceso a un directorio a varias horas concretas

```
# bloquear acceso en varias horas
RewriteCond %{TIME_HOUR} ^(12|13|14|15)$
RewriteRule ^.*$ - [F,L]
```

Denegar acceso a muchos proxies

```
# bloqueo de Proxies
RewriteCond %{HTTP:VIA} !^$ [OR]
RewriteCond %{HTTP:FORWARDED} !^$ [OR]
RewriteCond %{HTTP:USERAGENT_VIA} !^$ [OR]
RewriteCond %{HTTP:X_FORWARDED_FOR} !^$ [OR]
RewriteCond %{HTTP:PROXY_CONNECTION} !^$ [OR]
RewriteCond %{HTTP:XPROXY_CONNECTION} !^$ [OR]
RewriteCond %{HTTP:HTTP_PC_REMOTE_ADDR} !^$ [OR]
RewriteCond %{HTTP:HTTP_CLIENT_IP} !^$
RewriteRule ^(.*)$ - [F]
```

Prevenir accesos al archivo .htaccess

prevenir acceso .htaccess

```
<files .htaccess>
  order allow,deny
  deny from all
</files>
```

Prevenir el acceso a un archivo específico

```
#prevenir acceso a un fichero concreto
<files nombre.jpg>
  order allow,deny
  deny from all
</files>
```

Prevenir el acceso a varios tipos de archivos

```
# prevenir acceso a tipos de ficheros concretos
<FilesMatch "\.(htaccess|htpasswd|ini|phps|log)$">
  order allow,deny
  deny from all
</FilesMatch>
```

Evitar que se muestren determinados tipos de ficheros

```
# evitar muestra de fichero MP4, WMV y AVI
IndexIgnore *.wmv *.mp4 *.avi
```

Evitar el listado de directorios

```
# evitar listado de directorios
IndexIgnore *
```

Evitar el hotlinking en el dominio

El hotlinking es cuando descargan ficheros de tu servidor desde otra web para mostrarlos en ésta, 'robando' ancho de banda y servicios del tuyo.

```
# Evita el hotlinking a ficheros GIF y JPG del servidor
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://(www\.)?midominio\.com/.*$ [NC]
RewriteRule .*\.(gif|jpg)$ http://www.midominio.com/hotlinking.jpg [R,NC,L]
```

Nota: Es más eficaz, si el archivo que mostramos a aquellos que realizan el hotlinking lo colgamos en un servidor gratuito de imágenes tipo ImageShack.us (o similares) y lo referenciamos allí.

Bloqueo de ficheros determinados el cualquier subdominio, para evitar hotlinking específico

Para evitar el hotlinking a ficheros MP3 y ficheros de vídeo AVI, WMV y MPG:

```
# Evita el hotlinking a ficheros MP3, AVI, WMV y MPG de cualquier subdominio
y dominio del servidor
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://([-a-z0-9]+\.)?midominio\.com [NC]
RewriteRule .*\.(mp3|avi|wmv|mpg|mpeg)$
http://www.midominio.com/images/nohotlink.gif [R,NC,L]
```

Nota: si se envían ficheros por RSS -sindicación del sitio- debe tenerse en cuenta, ya que las instrucciones anteriores bloquean todo el contenido de vídeo y los RSS pueden verse mal.

Nota: Es más eficaz, si el archivo que mostramos a aquellos que realizan el hotlinking lo colgamos en un servidor gratuito de imágenes tipo ImageShack.us o similar, y lo referenciamos allí.

Bloqueo de robot -bot- específico, a través de SetEnvIfNoCase

```
# bloqueo del bot BotMalo
SetEnvIfNoCase User-Agent "BotMalo/" spambot
deny from env=spambot
```

Bloque de robot -bot- específico, a través de Rewrite

```
# bloqueo de 3 bots conocidos (hay muchísimos más)
RewriteCond %{HTTP_USER_AGENT} ^WWWOFFLE [OR]
RewriteCond %{HTTP_USER_AGENT} ^Xaldon\ WebSpider [OR]
RewriteCond %{HTTP_USER_AGENT} ^Zeus
RewriteRule .* - [F]
```

Bloqueo de robot -bot- que accede siempre al mismo fichero

```
# bloqueo del bot al acceder a un fichero PHP concreto
SetEnvIfNoCase Request_URI "/firefoxz.php$" spambot
deny from env=spambot
```

Bloqueo de robot -bot- que siempre es referenciado desde un mismo sitio, a través de SetEnvIfNoCase

```
# bloqueo cuando se viene referenciado de un sitio web concreto
SetEnvIfNoCase Referer "^http://www.dominiospammers.com/" spambot
deny from env=spambot
```

Nota: Las 3 anteriores sentencias pueden combinarse escribiendo las 3 líneas SetEnvIfNoCase, y luego solo un deny, de la siguiente forma:

```
SetEnvIfNoCase User-Agent "BotMalo/" spambot
SetEnvIfNoCase Request_URI "/firefoxz.php$" spambot
SetEnvIfNoCase Referer "^http://www.spammers.com/" spambot
deny from env=spambot
```

Bloqueo de robot -bot- que siempre es referenciado desde un mismo sitio, a través de Rewrite

```
# bloqueo cuando se viene referenciado de un sitio web concreto ejemplo1.com
RewriteCond %{HTTP_REFERER} ^ejemplo1\.com$ [NC]
RewriteRule .* - [F]
```

Bloqueo de robot -bot- que siempre es referenciado desde dos sitios, a través de Rewrite

```
# bloqueo cuando se viene referenciado desde ejemplo1.com o ejemplo2.com
RewriteCond %{HTTP_REFERER} ^ejemplo1\.com$ [NC,OR]
RewriteCond %{HTTP_REFERER} ^ejemplo2\.com$ [NC]
RewriteRule .* - [F]
```

Evitar acceso mediante el navegador a una carpeta sin el archivo “index”

```
# evitar acceso a carpeta sin index
Options All -Indexes
```

Nota: lo contrario, es decir, permitir el acceso a carpetas sin index es:

```
# evitar acceso a carpeta sin index
Options All +Indexes
```

3.- Re-escritura y redirección

Quitar siempre las www de la URL

Es importante no repetir URLs -mostrando el mismo contenido con y sin www-, ya que penaliza el posicionamiento.

```
# quitar las www
RewriteCond %{http_host} ^www\.netandsoftware\.com [NC]
RewriteRule ^(.*)$ http://netandsoftware.com/$1 [R=301,L]
```

Mostrar siempre las www de la URL

```
# mostrar siempre las www
RewriteCond %{HTTP_HOST} .
RewriteCond %{HTTP_HOST} !^www\.netandsoftware\.com$
RewriteRule (.*) http://www.netandsoftware.com/$1 [R=301,L]
```

Cambiar la página de carga por defecto

```
#orden de los ficheros de carga por defecto
DirectoryIndex inicio.html index.htm index.html index.php
```

Nota: El orden es muy importante porque da la preferencia de los ficheros de carga

Camuflar el tipo de ficheros

Esto se hace para que sea más dificil detectar el tipo de ficheros que se usa

```
# camuflar archivos PHP como NAS
AddType application/x-httpd-php .nas
```

Redirigir el contenido en función del navegador usado

```
RewriteCond %{HTTP_USER_AGENT} ^Opera/*
RewriteRule ^index\.html$ index.opera.html [L]
RewriteCond %{HTTP_USER_AGENT} ^Mozilla/*
RewriteRule ^index\.html$ index.mozilla.html [L]
```

```
RewriteRule ^index\.html$ index.html [L]
```

Nota: Los asteriscos indican el uso de cualquier versión de ese navegador, por ejemplo:dicarse a continuación de la barra, como “Mozilla/3.0”, “Mozilla/4.0”, “Mozilla/5.0”, etc.

Redireccionar una antigua página a la nueva

```
# redireccionar permanentemente
RewriteRule ^(.*)$ http://www.minuevodominio.com/$1 [R=301,L]
```

Redireccionar una antigua página a la nueva, a través de ReDirect

```
# redireccionar permanentemente
redirect 301 / http://www.dominio.com/
```

Redireccionar una web a otro sitio provisionalmente (redirección 302)

```
# redireccionar provisionalmente
RewriteRule ^(.*)$ http://www.minuevodominio.com/$1 [R=302,L]
```

Redireccionar de un archivo a otro

```
# redireccionar un antiguo fichero a otra nueva dirección
Redirect /antiguo.html http://dominio.com/nuevo.html
```

Redireccionar una IP concreta a una página concreta

Puede servir para avisar a una persona con una IP concreta de que ha sido baneada.

redireccionar una IP a una página concreta de nuestra web

```
RewriteCond %{REMOTE_ADDR} 100.101.102.103
RewriteRule .* pagina-concreta.html [R]
```

Redireccionar toda una web a un directorio del mismo dominio

```
# traslada todas las URL antiguas a la nueva carpeta en el mismo dominio
RewriteCond %{HTTP_HOST} ^midominio\.com$ [OR]
RewriteCond %{HTTP_HOST} ^www\.midominio\.com$
RewriteCond %{REQUEST_URI} !^/webencarpeta/
RewriteRule (.*) /webencarpeta/$1 [L]
```

Quitar una palabra de una URL

En el siguiente ejemplo, de http://www.midominio.com/quitar/prueba.html a http://www.midominio.com/prueba.html

```
# quitar una cadena de una URL
RewriteRule ^quitar/(.+)$ http://www.midominio.com/$1 [R=301,L]
```

URL amigable (uso de fin de interrogación en RewriteRule)

Cambia una URL como http://www.midominio.com/articulos-blog?catid=27 a http://www.midominio.com/articulos-blog de forma permanente.

```
# Uso de ?
RewriteCond %{REQUEST_URI} ^/articulos-blog$ [NC]
RewriteCond %{QUERY_STRING} ^catid=(.*)$ [NC]
RewriteRule ^(.*)$ /articulos-blog? [R=301,L]
```

URL amigable (uso del tanto por ciento en RewriteRule) Cambia una URL como http://www.dominio.com/noticias?id=127 a http://www.dominio.com/noticias/127 de forma provisional.

```
# Uso del %
RewriteCond %{REQUEST_URI} ^/noticias$ [NC]
RewriteCond %{QUERY_STRING} ^id=(.*)$ [NC]
RewriteRule ^(.*)$ /noticias/%1? [R=302,L]
```

Mejorando el posicionamiento SEO con RewriteRule Cambia una URL con la cadena “ps” por una URL con la cadena “posicionamiento-seo” que determina mejor el contenido de las URLs.

```
# mejorando el posicionamiento SEO sustituyendo caracteres sin sentido por
otros que determinan el contenido
RewriteRule ^(.*)/ps/(.*)$ $1/posicionamiento-seo/$2 [L,R=301]
```

Transformación de una URL con agrupamiento de caracteres mayúsculas y minúsculas

Cambia la URL de este tipo http://www.midominio.com/pais/Espana.php a http://www.midominio.com/codigo/pais.php?nombre=Espana

```
# detectar palabras en mayúsculas o minúsculas o combinacion de ambas
RewriteRule ^pais/([a-zA-Z_-]+).php$ codigo/pais.php?nombre=$1 [L]
```

Paso de parámetros en la URL y doble parámetro

Cambia una URL como http://www.midominio.com/fecha/2013/12.html a http://www.midominio.com/fecha.php?mes=12&anio=2013 de forma permanente

```
# Doble parámetro
RewriteRule ^fecha/(.+)/(.+)\.html$ fecha.php?mes=$2&anio=$1 [R=301,L]
```

4.- Compresión y caché

Comprimir ficheros de texto, HTML, JavaScript, CSS y XML

```
# comprimir focheros texto, html, javascript, css, xml
AddOutputFilterByType DEFLATE text/plain
AddOutputFilterByType DEFLATE text/html
AddOutputFilterByType DEFLATE text/xml
```

```
AddOutputFilterByType DEFLATE text/css
AddOutputFilterByType DEFLATE application/xml
AddOutputFilterByType DEFLATE application/xhtml+xml
AddOutputFilterByType DEFLATE application/rss+xml
AddOutputFilterByType DEFLATE application/javascript
AddOutputFilterByType DEFLATE application/x-javascript
```

También en una sóla línea: AddOutputFilterByType DEFLATE text/plain text/html text/xml text/css application/xml application/xhtml+xml application/rss+xml application/javascript application/x-javascript

Cachear ficheros de imágenes a una semana

```
#Cache del navegador, imagenes cacheadas a una semana: 604800 segundos
<FilesMatch "\.(ico|jpg|jpeg|png|gif)$">
  Header set Cache-Control "max-age=604800, public"
</FilesMatch>
```

Cachear ficheros JPG a un mes

```
#Cache del navegador, imágenes cacheadas a un mes: 2592000 segundos
<FilesMatch "\.(jpg|jpeg)$">
  Header set Cache-Control "max-age=2592000, public"
</FilesMatch>
```

Cachear con ExpiresByType y el intervalo legible

```
#caché que expira las imágenes JPG en 6 meses, los CSS en 2 meses, y los
JavaScripts en 2 semanas
ExpiresActive on
ExpiresByType image/jpg "access plus 6 months"
ExpiresByType text/css "access plus 2 months"
ExpiresByType text/javascript "access plus 2 weeks"
```

Cachear ficheros a un año con ExpiresDefault y el intervalo en segundos

```
ExpiresActive On
ExpiresDefault A0
#  caché expira en un año (A9030400) para los ficheros FLV, ICO, AVI, MOV,
PPT, DOC, MP3, WMV y WAV
<FilesMatch "\.(flv|ico|pdf|avi|mov|ppt|doc|mp3|wmv|wav)$">
  ExpiresDefault A9030400
</FilesMatch>
```

5.- Otros funcionalidades en el .htaccess

Asignar permisos CHMOD automáticamente al fichero .htpasswd

```
# permiso automático del fichero que protege el directorio .htpasswd
chmod .htpasswd files 640
```

Asignar permisos CHMOD automáticamente al fichero .htaccess

```
# permiso automático del fichero .htaccess
chmod .htaccess files 644
```

Asignar permisos CHMOD automáticamente a ficheros PHP

```
# permisos automáticos de archivos PHP
chmod php files 600
```

Limitar la subida de ficheros más grandes que una cantidad

Protege de ciertos ataques DOS, limitando el tamaño de archivos que se suben al servidor.

```
# limitar la subida a 10 MB
LimitRequestBody 10000000
```

Permitir al usuario descargar archivos multimedia

Generalmente sólo se permiten abrir

```
# permitir descarga de ficheros multimedia AVI, MPG, WMV y MP3
AddType application/octet-stream .avi
AddType application/octet-stream .mpg
AddType application/octet-stream .wmv
AddType application/octet-stream .mp3
```

Personalizar errores 404

```
ErrorDocument 404 /errores/404.html
```

Nota: Para otros tipos de errores es similar, se coloca el número de error y la dirección donde queremos redireccionar al visitante.

Corregir pequeños errores de ortografía en las URL

```
CheckSpelling On
```

Especificar el e-mail por defecto del administrador del servidor

```
# e-mail del administrador del servidor
SetEnv SERVER_ADMIN webmaster@midominio.com
```

Especificar el lenguaje por defecto del servidor

```
# lenguaje por defecto del servidor
DefaultLanguage en-US
```

Forzar el uso del protocolo seguro SSL

```
# fuerza el uso de SSL en la web
SSLOptions + StrictRequire
SSLRequireSSL
```

Redireccionar usuarios con protocolo seguro HTTPS a una carpeta en particular Esto se puede necesitar cuando una web tiene una tienda online en una carpeta específica

```
RewriteCond %{SERVER_PORT} 80
RewriteCond %{REQUEST_URI} carpeta
RewriteRule ^(.*)$ https://www.midominio.com/carpeta/$1 [R,L]
```

Evitar la visualización de errores al visitante

```
# evitar que los errores se muestren al usuario
php_flag display_startup_errors off
php_flag display_errors off
php_flag html_errors off
```

Registrar errores de PHP en un fichero log

```
# registrar errores en log
php_flag log_errors on
php_value error_log /logs/php_error.log
```

Limitar el número de visitas al mismo tiempo a 400

```
# limitar el número de visitas a 400
MaxClients 400
```

Denegar la ejecución de scripts CGI

```
# bloquear CGIs
Options -ExecCGI
AddHandler cgi-script .php .pl .py .jsp .asp .sh .cgi
```

Declarar tipos MIME

```
# agregar tipos mime
AddType application/x-shockwave-flash .swf
AddType video/x-flv .flv
AddType image/x-icon .ico
```

Definiendo el juego de caracteres

Pasando el juego de caracteres se evita el mostrar un error 500 por este motivo

```
AddDefaultCharset utf-8
```

Desactivar los Entities Tags

```
FileETag none
```

6.- Recomendaciones sobre el fichero .htaccess

El tamaño es importante: cuanto más pequeño sea el fichero .htaccess menos tiempo tardará el servidor en procesarlo cada vez que se realiza una petición a tu servidor. Esto implica una pérdida de rendimiento en la carga de las páginas de tu sitio web si este fichero se hace demasiado grande.

La directiva [L] es realmente interesante y hace que el servidor no procese más el archivo una vez que se cumple esa regla. Por lo que incorpora siempre que puedas esta directiva en los RewriteRule. La organización es fundamental: dado lo complicada que pueden ser ciertas líneas en el código de .htaccess, es imprescindible comentar adecuadamente tu fichero para que la modificación sea rápida y sencilla. No escatimes en comentarios, y organiza adecuadamente las instrucciones de tu .htaccess. Protege adecuadamente este archivo de reescrituras ajenas, puede llegar a ser muy peligroso.

La denegación de permisos debe estar lo primero, antes de ejecutar RewriteCond y RewriteRule.

7.- Simbología del fichero .htaccess

De forma breve, los códigos y símbolos en el archivo .htaccess son:

Puesto al inicio de una línea, ignora esa línea.

[F] Forbidden: prohíbe un acceso y fuerza un acceso denegado. 403 Forbidden.

[L] Last rule: indica que es la última regla que debe aplicarse.

[N] Next: indica continuación hasta que las directivas sean logradas.

[G] Gone: indica al servidor que ya no existe, es decir, entrega “Gone”.

[P] Proxy: instruye al servidor para manejar los pedidos por mod_proxy.

[C] Chain: encadena la regla actual con la regla anterior.

[R] Redirect: indica redirección. Puede haber de varios tipos 301 (permanente), 302 (provisional).

[NC] No Case: no sensible a mayúsculas, es decir, indica que no debe distinguirse entre mayúsculas y minúsculas.

[PT] Pass Through: pasa el URL a Apache para seguir procesando.

[OR] Or: indica que la expresión debe interpretase como una alternativa junto a la siguiente: o lógico.

[M] para definir un tipo mime. Ejemplo: “[M]=text/plain”.
[P] para definir un handler que procese la petición. Ejemplo: “[P]=/cgi-bin/procesa.cgi”.
[PT] Pass Through: pasa la petición a la siguiente regla. Si se pone este símbolo al final del segundo argumento del RewriteRule, indicará que no se ponga nada más en la URL.
[R=xxx] Redirect: redirige a la URL indicada. “xxx” es el código de redirección.
[S=x] Skip: salta las siguientes “x” reglas del fichero .htaccess.
[T=xxx] Type: para definir el tipo mime. Ejemplo: “[T=text/plain]”.

Si se omite, se sobreentiende que es una Y lógica, por defecto.
Si se pone un signo de exclamación delante de una cadena, indica que esa cadena no tiene valor cero.
[NE] No Escape: analiza las salidas de caracteres sin escapar.
! es la negación. Ejemplo: “!string” resulta “no string”.

[NS] No Subrequest: para saltar directivas de sub-pedidos internos.
. indica cualquier carácter arbitrario.

[QSA] Append Query String: agrega un query string al final de la expresión (URL).
- instrucción “not to”.

From:
https://wiki.merkatu.info/ - **Wiki de merkatu**
Permanent link:
https://wiki.merkatu.info/htaccess_en_profundidad?rev=1424689333
Last update: **2017/03/27 17:43**

+ indica uno o más caracteres del carácter que le precede. Por ejemplo: (.+) indica cualquier cadena de una o más caracteres.
[E=variable:valor] Environmental Variable: para añadir un valor a una variable.
| es el lógico, condición de alternancia. Ejemplo (a|b) a o b.
[H=xxx] Handler: para definir el handler del recurso.

\ dentro de los paréntesis se encierran caracteres que definen los resultados.
\ escapa caracteres, es decir, toma el caracter que le sigue literalmente. Por ejemplo: “\.” indica literalmente un punto, ya que sin la barra invertida indicaría cualquier caracter.
[]+ se utiliza para hacer combinaciones.
/* indica cero o más “/”.
[^] excluye los carácteres que pongamos dentro del paréntesis. Ejemplo [^abc] excluye las letras a, b y c.
.* indica cualquier caracter, incluido que no exista caracter.

[a-z] letras desde la a hasta la z (en minúsculas).

[A-Z] letras desde la A hasta la Z (en mayúsculas).

[a-zA-Z] sólo letras (mayúsculas y minúsculas).

[a-z]{1,10} palabras en minúsculas entre 1 y 10 caracteres.

[0-9] sólo un número.

[0-9]{4} números de 4 cifras.

(.*) cualquier expresión, incluida la vacía.

a{n} especifica el número de caracteres.

a{n,} especifica el número “o más” de caracteres.

a{n,m} especifica un rango entre “n” y “m”. Ejemplo s{3,6} será 3 “eses”, 4 “eses”, 5 “eses” o 6 “eses”.

() es un agrupamiento de caracteres.