EBook Gratis

APRENDIZAJE

Hypertext Access file

Free unaffiliated eBook created from
Stack Overflow contributors.

#.htaccess

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: hypertext-access-file

It is an unofficial and free Hypertext Access file ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Hypertext Access file.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Comenzando con el archivo de acceso de hipertexto

Observaciones

Un archivo `.htaccess` controla la forma en que Apache interactúa con su sitio. Cuando se coloca un archivo `.htaccess` en el directorio de su dominio (generalmente el directorio raíz), Apache detecta y ejecuta el archivo.

Un archivo `.htaccess` se usa comúnmente para lo siguiente:

- Denegar direcciones IP específicas a su sitio
- Contraseña protegiendo su sitio
- Reescribiendo URLs
- Páginas de error personalizadas
- Archivos de compresión y almacenamiento en caché
- Seguridad general y prevención de hackers

Versiones

Varios lanzamientos de apache

Versión	Versión actual	Lanzamiento
1.3	1.3.42	1998-06-06
2.0	2.0.65	2002-04-06
2.2	2.2.31	2005-12-01
2.4	2.4.23	2012-02-21

Examples

Configurando .htaccess

> `.htaccess` archivos `.htaccess` (o "archivos de configuración distribuidos") proporcionan una forma de realizar cambios de configuración por directorio. Un archivo, que contiene una o más directivas de configuración, se coloca en un directorio de documentos en particular, y las directivas se aplican a ese directorio, y todos sus subdirectorios.

Un archivo .htaccess controla la forma en que Apache interactúa con su sitio. Se utiliza para modificar las solicitudes y modificar el comportamiento predeterminado sin necesidad de alterar

los archivos de configuración del servidor central.

Configurar `.htaccess` es tan simple como abrir una libreta y guardarla como `.htaccess` . En general, este archivo se colocará en el directorio `root` de los archivos de su sitio web, pero puede usarlo en varios directorios diferentes. Esto es especialmente útil si está buscando proteger con contraseña directorios específicos.

Habilitando .htaccess

> A veces, incluso un solo error en el archivo `httpd.conf` o `.htacces` s provocará un colapso temporal del servidor, y los usuarios verán *500* - Página de *error interno del servidor* . Por lo tanto, asegúrese de hacer siempre una copia de seguridad de sus archivos `httpd.conf` y `.htaccess` antes de realizar un cambio.

```
<Directory "/var/www">
    AllowOverride All
</Directory>
```

`.htaccess` archivos `.htaccess` normalmente están habilitados por defecto. Esto se controla mediante la directiva `AllowOverride` en el archivo `httpd.conf` . Esta directiva solo se puede colocar dentro de una sección `<Directory>` .

Además de `All` hay muchos otros valores que limitan la configuración de solo ciertos contextos. Algunos de ellos son:

- **Ninguno** - Deshabilita completamente `.htaccess` .
- **AuthConfig** : directivas de autorización, como las relacionadas con la autenticación básica.
- **FileInfo** : directivas relacionadas con la configuración de encabezados, documentos de error, cookies, reescritura de URL y más.
- **Índices** - Personalización predeterminada de listas de directorios.
- **Límite** : controle el acceso a las páginas de diferentes maneras.
- **Opciones** : acceso similar a los índices, pero incluye incluso más valores como ExecCGI, FollowSymLinks, Incluye y más.

```
# Only allow .htaccess files to override Authorization and Indexes
AllowOverride AuthConfig Indexes
```

Páginas de error personalizadas

`.htaccess` se puede usar para establecer páginas de error personalizadas que coincidan con el tema de su sitio web, en lugar de ver una página de error en blanco con balbuceo negro cuando los usuarios terminan en una página con un código de respuesta del servidor de errores. La página de error puede ser cualquier archivo analizable del navegador, incluido (pero no limitado a) .html, .php, .asp, .txt, .xml.

Ejemplos para casi todos los códigos de respuesta de error comunes:

```
#Client Errors

ErrorDocument 400 /mycool400page.html    # Bad Request
ErrorDocument 401 /mycool401page.html    # Unauthorized
ErrorDocument 402 /mycool402page.html    # Payment Required
ErrorDocument 403 /mycool403page.html    # Forbidden
ErrorDocument 404 /mycool404page.html    # Page Not Found

#Server Errors

ErrorDocument 500 /mycool500page.html    # Internal Server Error
ErrorDocument 501 /mycool501page.html    # Not Implemented
ErrorDocument 502 /mycool502page.html    # Bad Gateway
ErrorDocument 503 /mycool503page.html    # Service Unavailable
ErrorDocument 504 /mycool504page.html    # Gateway Timeout
ErrorDocument 505 /mycool505page.html    # Internal Server Error
```

Siempre es una buena práctica incluir Documentos de error para las respuestas de error más comunes, 400, 403, 404 y 500, ya que estos errores pueden ocurrir en todos los navegadores.

el error 500 es uno de los errores más notorios, ya que ocurre si algo falla al cargar la página para enviar, con mayor frecuencia servidores de preprocesamiento de html de cosas como PHP, ASP y otros preprocesadores de html. Es una buena práctica, mientras se prueba, configurar la página 500 para mostrar el error que se produjo, en lugar de una página de error 500 no específica.

Para habilitar la página de error 500 para escribir un error específico, consulte una de las siguientes opciones según el preprocesador html que esté utilizando: php asp

Configuración de la zona horaria del servidor

Hay muchas zonas horarias en todo el mundo, es importante asegurarse de que su servidor esté configurado en la correcta. Esto se hace en `.htaccess` usando:

```
SetEnv TZ America/Indianapolis
```

Algunos ejemplos de otras posibles zonas horarias:

```
America/Los_Angeles
America/Los_Angeles - Pacific Time
Pacific/Honolulu - Hawaii
```

Solo asegúrate de usar `SetEnv` delante de tu zona horaria seleccionada.

Lea Comenzando con el archivo de acceso de hipertexto en línea: https://riptutorial.com/es/dot-htaccess/topic/1023/comenzando-con-el-archivo-de-acceso-de-hipertexto

Capítulo 2: Manejo de tipos de archivos

Examples

Habilitar PHP para ser analizado en HTML

Si desea incluir el código PHP en su archivo HTML y no desea cambiar el nombre del tipo de archivo de `.html` o `.htm` a `.php` , la siguiente información permite que su archivo HTML analice su código PHP correctamente.

```
AddHandler application/x-httpd-php .html .htm
```

Lea Manejo de tipos de archivos en línea: https://riptutorial.com/es/dot-htaccess/topic/1690/manejo-de-tipos-de-archivos

Capítulo 3: Negando el acceso

Examples

Negando IPs

```
order allow,deny
deny from 255.0.0.0
allow from all
```

Esto niega el acceso a la IP `255.0.0.0` .

```
order allow,deny
deny from 123.45.6.
allow from all
```

Esto niega el acceso a todas las IP en el rango de `123.45.6.0` a `123.45.6.255` .

Prevención de enlaces calientes

```
RewriteEngine on
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://(www\.)?yourdomain.com/.*$ [NC]
RewriteRule \.(gif|jpg|css)$ - [F]
```

Esto bloquea todos los enlaces a los archivos '.gif', '.jpg' y '.css' que no pertenecen al nombre de dominio `http://www.yourdomain.com` .

Mostrar contenido alternativo:

```
RewriteEngine on
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://(www\.)?yourdomain.com/.*$ [NC]
RewriteRule \.(gif|jpg)$ http://www.yourdomain.com/angryman.jpg [R,L]
```

Esto bloquea todos los enlaces a los archivos '.gif' y '.jpg' que no pertenecen al nombre de dominio ' http://www.yourdomain.com/ ' y muestra el archivo ' http://www.yourdomain.com/angryman ' .jpg 'en su lugar.

Denegación de acceso de direcciones IP a archivos / directorios

```
# Deny access to a directory from the IP 255.0.0.0
<Directory /path/to/directory>
    order allow,deny
    deny from 255.0.0.0
    allow from all
</Directory>
```

```
# Deny access to a file from the IP 255.0.0.0
<FilesMatch "^\.ht">
    order allow,deny
    deny from 255.0.0.0
    allow from all
</FilesMatch>
```

Lea Negando el acceso en línea: https://riptutorial.com/es/dot-htaccess/topic/4741/negando-el-acceso

Capítulo 4: Optimización de la velocidad

Examples

Habilitar la compresión (Apache 2.0+)

> Habilitar la compresión gzip puede reducir el tamaño de la respuesta transferida hasta en un 90%, lo que puede reducir significativamente la cantidad de tiempo para descargar el recurso, reducir el uso de datos para el cliente y mejorar el tiempo para la primera representación de sus páginas. - *PageSpeed Insights*

La compresión se puede habilitar con esto:

```
AddOutputFilterByType DEFLATE "text/html"/
                              "text/plain"/
                              "text/xml"/
                              "text/css"/
                              "text/javascript"/
                              "application/javascript"
```

Apache Docs

Aproveche el almacenamiento en caché del navegador (Apache 2.0+)

> La obtención de recursos a través de la red es lenta y costosa: la descarga puede requerir múltiples viajes de ida y vuelta entre el cliente y el servidor, lo que retrasa el procesamiento y puede bloquear el procesamiento del contenido de la página, y también incurrir en costos de datos para el visitante. Todas las respuestas del servidor deben especificar una política de almacenamiento en caché para ayudar al cliente a determinar si y cuándo puede reutilizar una respuesta obtenida previamente. - *PageSpeed Insights*

Puede aprovechar el almacenamiento en caché del navegador de esta manera:

```
# Enable browser caching
ExpiresActive On

# Set the default caching duration
ExpiresDefault "access plus 1 week"

# Change the caching duration by file type
ExpiresByType text/html "access plus 2 weeks"
```

Apache Docs

Habilitar KeepAlive (Apache 2.0+)

> La extensión Keep-Alive a HTTP / 1.0 y la característica de conexión persistente de

> HTTP / 1.1 proporcionan sesiones HTTP de larga duración que permiten enviar múltiples solicitudes a través de la misma conexión TCP. En algunos casos, se ha demostrado que esto da como resultado una aceleración de casi el 50% en los tiempos de latencia para documentos HTML con muchas imágenes. Para habilitar las conexiones Keep-Alive, active KeepAlive. - *Apache Docs*

```
# Enable KeepAlive
KeepAlive On

# OPTIONAL — limit the amount of requests per connection with 'MaxKeepAliveRequests'
# Example: MaxKeepAliveRequests 500

# OPTIONAL — limit the amount of time the server will wait before it closes
# the connection with 'KeepAliveTimeout'
# Example: KeepAliveTimeout 500
```

Apache Docs

Lea Optimización de la velocidad en línea: https://riptutorial.com/es/dot-htaccess/topic/3893/optimizacion-de-la-velocidad

Capítulo 5: Reescritura y redireccionamiento

Observaciones

Antes de poder reescribir las URL, se debe habilitar un módulo llamado `mod_rewrite.c` . Por lo general, está deshabilitado en la configuración por defecto.

`mod_rewrite` se puede habilitar ejecutando el comando

```
$ sudo a2enmod mod_rewrite
$ sudo service apache2 restart
```

o comentando las líneas

```
#LoadModule rewrite_module modules/mod_rewrite.so
#AddModule mod_rewrite.c
```

en el archivo `httpd.conf`

Examples

Banderas de reescritura populares

F | prohibido

Similar a `Deny` , esta bandera obliga al servidor a devolver inmediatamente un código de estado *403 Prohibido* al navegador o cliente que solicita la solicitud.

Ejemplo: Denegar el acceso a solicitudes que terminan con `exe` :

```
RewriteRule .exe$ - [F]
```

G | ido

Si un recurso solicitado estaba disponible en el pasado, pero ya no está disponible, puede usar este indicador para forzar al servidor a devolver inmediatamente un código de estado *410 Gone* al navegador o cliente que solicita la solicitud.

Ejemplo: Dígale a un visitante que un producto antiguo ya no existe:

```
RewriteRule ^old-product.html$ - [G]
```

L | ultima

En la mayoría de los contextos, aparte de `.htaccess` , este indicador indica a `mod_rewrite` que deje de procesar el conjunto de reglas / condición actual, de la misma forma que lo hace `last` y `break` (Perl y C, respectivamente).

Sin embargo, en el contexto `.htaccess` o `<Directory>` , una solicitud que se haya reescrito utilizando una `RewriteRule` con esta bandera se pasará de nuevo al motor de análisis de URL para su posterior procesamiento. Como tal, es posible que la URI reescrita se maneje en el mismo contexto, y quizás se modifique más.

Una recomendación general es utilizar el indicador `END` no solo para detener el procesamiento de la condición / conjunto de reglas actual, sino también para evitar que se vuelva a escribir en estos contextos.

Nota: los indicadores `F` y `G` , discutidos anteriormente, usan `L` implícitamente, por lo que no es necesario especificarlos por separado.

N | siguiente

Este indicador volverá a ejecutar el proceso de reescritura desde el principio, comenzando nuevamente con el primer conjunto de condiciones / reglas. Esta vez, la URL a coincidir ya no es el URI original, sino el URI reescrito devuelto por el último conjunto de reglas. Utilice este indicador para reiniciar el proceso de reescritura.

Una advertencia: use esta bandera con precaución, ya que puede dar lugar a un bucle infinito.

NC | nocase

Esto indica a `mod_rewrite` que coincida con el `Pattern` de una `RewriteRule` sin `RewriteRule` mayúsculas y minúsculas. Para aclarar, `MyIndex.html` y `myindex.html` serían considerados por el módulo como la misma cosa. Además, esta bandera le permite usar `az` lugar de `A-Za-z` en una expresión regular.

R | redireccionar

Este indicador se utiliza para enviar una respuesta de redireccionamiento HTTP al navegador / cliente solicitante.

De forma predeterminada, si no se proporciona ningún código, se devolverá una respuesta de redireccionamiento con el código de estado *302 Encontrado* (similar a un redireccionamiento temporal). Si desea utilizar una redirección más permanente, debe utilizar el código de estado `302` (*301 Movido permanentemente*).

Generalmente, solo los códigos de estado en el rango 300-399 deben usarse con esta bandera. Si se utilizan códigos de estado fuera de este rango (lo que es perfectamente aceptable), la cadena de sustitución se descarta y la reescritura se detiene como si se usara la bandera `L` En algunos casos, esta es una forma útil de forzar las respuestas *404 No Encontradas* , incluso si la solicitud apunta a un recurso existente.

Ejemplo: Emita una respuesta de redirección *302 encontrada* :

```
RewriteRule ^bus$ /train [R,L]
```

Ejemplo: Emitir una respuesta de redireccionamiento *301 Movido permanentemente* :

```
RewriteRule ^speed-train$ /hyperloop [R=301,L]
```

Ejemplo: Forzar un *404 No encontrado* :

```
RewriteRule ^blip$ - [R=404,L]
```

www y redireccionamientos no www

Redirige cualquier dominio desnudo a `www.[your_domain].tld` :

```
# Start Apache Rewriting engine
RewriteEngine On
# Make sure you're not already using www subdomain
# and that the host string is not empty
RewriteCond %{HTTP_HOST} !^$
RewriteCond %{HTTP_HOST} !^www\.
# We check for http/https connection protocol
RewriteCond %{HTTPS}s ^on(s)|
# In case the previous conditions matches, redirect to www
RewriteRule ^(.*)$ http%1://www.%{HTTP_HOST}/$1 [R=301,L]
```

Redireccione `www.[your_domain].tld` a `[your_domain].tld`

```
# Start Apache Rewriting engine
RewriteEngine On
# We check if we're on the www subdomain
RewriteCond %{HTTP_HOST} ^www\.([^\.]+\.[^\.]+)$
# In case the previous condition matches, redirect to non-www
RewriteRule ^(.*)$ http://%1/$1 [R=301,L]
```

Redirige cualquier nivel de subdominios anidados a tu dominio principal:

```
# Start Apache Rewriting engine
RewriteEngine On
# We check if there's a subdomain
RewriteCond %{HTTP_HOST} \.([^.]+\.[^.]+)$
# redirect to the main domain name
RewriteRule ^ http://%1%{REQUEST_URI} [R=301,L]
```

URLs amigables para SEO

Los motores de búsqueda no indexarán sus productos si tiene una URL como la siguiente:

```
http://www.yourdomain.com/product.php?id=123
```

La URL amigable para SEO se vería como `http://www.yourdomain.com/123/product-name/` . El siguiente código ayuda a lograr esto sin tener que cambiar el código `product.php` .

```
RewriteEngine On
RewriteRule ^product/([0-9]+)/product-name-slug/?$ product.php?id=$1
```

Añadiendo una barra al final

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_URI} !(.*)/$
RewriteRule ^(.*)$ /$1/ [L,R=301]
```

El primer `RewriteCond` ayuda a excluir los archivos. El segundo `RewriteCond` comprueba si ya hay una barra diagonal. Si el caso es así, `RewriteRule` no se aplica.

Si tiene alguna URL que no deba ser reescrita, puede agregar un `RewriteCond` más.

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_URI} !(.*)/$
RewriteCond %{REQUEST_URI} !url/to/not/rewrite
RewriteRule ^(.*)$ /$1/ [L,R=301]
```

Redirecciones http y https y configuración HSTS

Redireccionamiento genérico *a https* :

```
# Enable Rewrite engine
RewriteEngine on

# Check if URL does not contain https
RewriteCond %{HTTPS} off [NC]
# If condition is true, redirect to https
RewriteRule (.*) https://%{SERVER_NAME}/$1 [R=301,L]
```

Redireccionamiento genérico *a http* :

```
# Enable Rewrite engine
RewriteEngine on

# Check if URL does contain https
RewriteCond %{HTTPS} on [NC]
# If condition is true, redirect to http
RewriteRule (.*) http://%{SERVER_NAME}/$1 [R=301,L]
```

Forzando la conexión HTTPS (HSTS):

```
<IfModule mod_headers.c>
    Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
</IfModule>
```

donde, la opción `includeSubDomains` se puede eliminar, si HSTS se debe aplicar solo al dominio base, o al dominio con la configuración anterior.

Redirigir con / sin parametros de consulta

Redireccionar sin parametros de consulta:

```
RewriteRule ^route$ /new_route_without_query [L,R=301,QSD]
```

Redireccionar con parámetros de consulta:

```
RewriteCond %{QUERY_STRING} ^$
RewriteRule ^/?route$ %{REQUEST_URI}?query=param1&query2=param2 [NC,L,R=301]
```

Lea Reescritura y redireccionamiento en línea: https://riptutorial.com/es/dot-htaccess/topic/1550/reescritura-y-redireccionamiento

Capítulo 6: Seguridad general y prevención de hackers

Observaciones

La redirección de .htaccess es un vector común para los hackers maliciosos para explotar e infectar sitios web. Hemos visto qué son los archivos .htaccess, cómo los utilizan los piratas informáticos malintencionados y cómo proteger su sitio web.

Examples

Prevención de Hack

Evite el acceso a su archivo `.htaccess`

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

# Rename the file
AccessFileName thehtfile.ess
```

Prevenir ataques de URL

```
# Enable rewrites
RewriteEngine On

# Block <script> tags from executing in the URL
RewriteCond %{QUERY_STRING} (<|%3C).*script.*(>|%3E) [NC,OR]

# Block scripts from setting a PHP Globals variable
RewriteCond %{QUERY_STRING} GLOBALS(=|[|\%[0-9A-Z]{0,2}) [OR]

# Block scripts from using base64_encode
RewriteCond %{QUERY_STRING} base64_encode.*(.*) [OR]

# Block scripts from using the a_REQUEST variable
RewriteCond %{QUERY_STRING} _REQUEST(=|[|\%[0-9A-Z]{0,2})
```

Deshabilite el uso de scripts en sus directorios.

```
AddHandler cgi-script .php .pl .py .jsp .asp .htm .shtml .sh .cgi
Options -ExecCGI
```

Deshabilitar índice de directorio

El índice de directorio habilitado significa que si alguien accede a cualquier carpeta que no contenga index.php, index.html, index.htm o cualquier otro archivo predeterminado definido en DirectoryIndex en la configuración de apache, todos los archivos de esa carpeta se mostrarán en el navegador si intentas visitar esa pagina

A menudo, el índice de directorio está habilitado por defecto en su servidor apache, en estos casos, una buena práctica de seguridad es deshabilitar el índice de directorio con la siguiente línea:

```
Options -Indexes
```

Lea Seguridad general y prevención de hackers en línea: https://riptutorial.com/es/dot-htaccess/topic/2531/seguridad-general-y-prevencion-de-hackers

Creditos

S. No	Capítulos	Contributors
1	Comenzando con el archivo de acceso de hipertexto	Community, Dilip Raj Baral, hjpotter92, James Oswald, Lag, Mike Rockétt, tbodt
2	Manejo de tipos de archivos	Dilip Raj Baral, John R Perry, Jon Lin, Marvin, mauris, Mike Rockétt, Nicholas Qiao
3	Negando el acceso	Dilip Raj Baral, John R Perry, tbodt
4	Optimización de la velocidad	John R Perry
5	Reescritura y redireccionamiento	Bogdan Alexandru Militaru, Dilip Raj Baral, Florian Lemaitre, hjpotter92, James, John R Perry, Mike Rockétt, shaN, Sven Reuter
6	Seguridad general y prevención de hackers	ban17, Dilip Raj Baral, John R Perry, Lag, Meysam, Mike Rockétt, OpenWebWar