EBook Gratis

APRENDIZAJE apache

Free unaffiliated eBook created from
Stack Overflow contributors.

#apache

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: apache

It is an unofficial and free apache 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 apache.

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: Empezando con apache

Observaciones

Esta sección proporciona una descripción general de qué es apache y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema grande dentro de apache, y vincular a los temas relacionados. Dado que la Documentación para apache es nueva, es posible que deba crear versiones iniciales de los temas relacionados.

Versiones

Varios lanzamientos de Apache httpd

Versión	Versión actual	Lanzamiento
1.3	1.3.42	1998-06-06
2.0	2.0.65	2002-04-06
2.2	2.2.32	2005-12-01
2.4	2.4.25	2012-02-21

Examples

Instalación o configuración

Instrucciones detalladas sobre cómo configurar o instalar apache.

Instalación de Ubuntu

```
sudo apt-get install apache2
```

Instalacion de windows

Echa un vistazo a la pila de WAMP . WAMP significa Windows, Apache, MySQL, PhpMyAdmin.

Instalacion centOS

Apache 2.2 viene con CentOS6, mientras que 2.4 viene con CentOS7, para instalar en cualquier sistema operativo, ejecute

```
yum -y install httpd
```

instalacion macOS

macOS viene con Apache preinstalado, sin embargo, puede instalar Apache a través de Homebrew

Si ya tiene el Apache integrado en ejecución, primero deberá apagarlo y se eliminarán todos los scripts de carga automática.

```
$ sudo apachectl stop
$ sudo launchctl unload -w /System/Library/LaunchDaemons/org.apache.httpd.plist 2>/dev/null
$ brew install httpd24 --with-privileged-ports --with-http2
```

[Ubuntu] Ejemplo simple de Hello World

Este ejemplo lo guiará a través de la configuración de un back-end que sirve una página HTML de Hello World.

Requisitos de instalación

¡La orden importa para este paso!

- `sudo apt-get install apache2`

Configurando el HTML

Los archivos de Apache viven en `/var/www/html/` . Vamos a llegar rápidamente. Asegúrese de estar primero en su directorio raíz, `cd` , luego `cd /var/www/html/` .

Este directorio `html` es donde vivirán todos los archivos de su sitio web. Vamos a hacer rápidamente un sencillo archivo Hello World.

Usando su editor de texto favorito, escriba lo siguiente en

```
<!DOCTYPE html>
<html>
<head>
    <title>Hello World!</title>
</head>
<body>
    <h1>Hello World!</h1>
</body>
```

```
</html>
```

Guarda este archivo como `index.html` en el directorio actual y listo!

Visitando tu pagina web

Para visitar la página que acaba de crear, en el navegador que elija, vaya a `localhost` . Si eso no funciona, intente `127.0.0.1` . Deberías ver "¡Hola mundo!" como un `h1` . Has terminado

Para asegurar que el servidor esté activo.

Si recibe un mensaje que indica que el navegador no puede conectarse al servidor, primero verifique que el servidor esté activo.

```
$ ps -aef | grep httpd
```

Debería ver algunos procesos httpd si Apache está en funcionamiento.

Lea Empezando con apache en línea: https://riptutorial.com/es/apache/topic/964/empezando-con-apache

Capítulo 2: Archivos .htaccess en Apache

Examples

Reescribir el motor

El módulo RewriteEngine dentro de Apache se usa para reescribir dinámicamente las URL y las rutas de acceso, dependiendo de las diversas expresiones proporcionadas:

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [END]
</IfModule>
```

Las reglas anteriores reescribirán los archivos PHP para que ya no muestren su extensión, y así index.php solo se mostrará como un dominio desnudo (similar al comportamiento que se ve normalmente en index.html). La regla anterior viene con WordPress.

Tenga en cuenta que en Apache httpd 2.2.16 y versiones posteriores, este bloque completo puede reemplazarse con una sola línea utilizando la directiva FallbackResource:

```
FallbackResource /index.php
```

Fuerza HTTPS

.htaccess puede usarse para forzar a su sitio HTTP a redireccionar a HTTPS.

Aquí hay una manera rápida que no requiere editar el código de su dominio:

```
RewriteEngine On
RewriteCond %{HTTPS} =off
RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]
```

> **Advertencia:** el código anterior asume que puede confiar en `%{HTTP_HOST}` para que apunte a su dominio.
>
> Si necesita asegurarse de que la ubicación de redireccionamiento sea su dominio, reemplace `%{HTTP_HOST}` con su dominio.

El código anterior hace esto:

1. Habilitar RewriteEngine .
2. Continuar si la solicitud actual no está utilizando HTTPS.
3. Realice una redirección HTTP 301 a `https://%{HTTP_HOST}%{REQUEST_URI}` , donde

- `%{HTTP_HOST}` es el host solicitado por el navegador y
- `%{REQUEST_URI}` es el URI solicitado por el navegador (todo lo que se encuentra después del dominio).

Advertencia: su aplicación web debe poder manejar solicitudes HTTPS, y Apache para su host debe configurarse con un certificado de sitio válido.

Tenga en cuenta que es significativamente más eficiente simplemente hacer una `Redirect` en el servidor de correo de http que hacer estas comparaciones múltiples por solicitud en un archivo .htaccess. Consulte http://wiki.apache.org/httpd/RedirectSSL para obtener más información sobre esta técnica.

Habilitar CORS

Para habilitar el uso ***compartido de recursos de origen cruzado*** (**CORS**) en Apache, deberá configurar al menos un encabezado HTTP que lo cambie (el comportamiento predeterminado es bloquear CORS). En el siguiente ejemplo, vamos a configurar este encabezado HTTP dentro de `.htaccess` , pero también se puede configurar en el archivo `your-site.conf` su sitio o el archivo de configuración de Apache. Independientemente de cómo se vea su configuración, puede configurar los encabezados HTTP relevantes en cualquier bloque de configuración de Apache, es decir, `<VirtualHost>` , `<Directory>` , `<Location>` y `<Files>` .

Hay algunos encabezados HTTP relacionados con CORS que puede devolver en la respuesta:

```
Access-Control-Allow-Origin

Access-Control-Allow-Credentials
Access-Control-Allow-Methods
Access-Control-Max-Age
Access-Control-Allow-Headers
Access-Control-Expose-Headers
```

Algunos de los anteriores son necesarios para las solicitudes de "verificación previa". Algunos clientes HTTP (es decir, los navegadores modernos) realizan una solicitud **antes de** la solicitud deseada solo para ver si tienen autorización para realizar la solicitud real en el servidor. Consulte https://en.wikipedia.org/wiki/Cross-origin_resource_sharing para obtener más información sobre la solicitud de verificación previa.

El encabezado HTTP principal que necesitamos es `Access-Control-Allow-Origin` y eso es lo que vamos a configurar. Sin embargo, el mismo principio se aplica a todos ellos (solo necesita saber qué devolver).

El siguiente ejemplo establece el encabezado HTTP requerido dentro de un bloque de configuración `<Directory>` para habilitar un Nombre de dominio completo calificado (FQDN) del cliente protegido por SSL:

```
<Directory /path/to/your/site/>
        Header set Access-Control-Allow-Origin "https://my.CLIENT.domain"
</Directory>
```

Después de haber configurado esto en el **servidor** , ahora podemos realizar una solicitud de https: //my.client.domain a nuestro servidor y debería responder.

Nota: Muchas personas usan `Access-Control-Allow-Origin: "*"` que es un comodín, para indicar que se deben aceptar solicitudes de **TODOS los** dominios. Por lo general, esto no es recomendable a menos que esté ejecutando algún tipo de API pública o repositorio de archivos. Además, tenga en cuenta el contexto de su configuración de encabezado HTTP. Es posible que desee permitir que las peticiones HTTP para una API, pero no para imágenes "hotlinking", etc. Puede configurar esta cabecera en cualquier lugar que desee dentro de su flujo de configuración de Apache que **sólo se** establece que en situaciones específicas. Por ejemplo, lo siguiente **solo** establecería el encabezado HTTP CORS cuando la ruta solicitada **no sea** un archivo o directorio (se adapta a una API pública que no permite el hotlinking de imágenes):

```
<Directory /path/to/your/site/>
    Options +FollowSymlinks
    Options +Indexes
    RewriteEngine On

    #Make sure it's not a specific file or directory that they're trying to reach
    RewriteCond %{SCRIPT_FILENAME} !-f
    RewriteCond %{SCRIPT_FILENAME} !-d
    Header set Access-Control-Allow-Origin "*"
    RewriteRule ^(.*)$ index.php/$1 [L]
</Directory>
```

Prerrequisitos

Tienes que tener mod_headers instalados y habilitados: `a2enmod headers`

301 Redireccionamiento por Htaccess

El código de estado de respuesta HTTP 301 Movido permanentemente se usa para la redirección permanente de URL, lo que significa que los enlaces o registros actuales que usan la URL para la cual se recibe la respuesta deben actualizarse. La nueva URL debe proporcionarse en el campo Ubicación incluido con la respuesta. La redirección 301 se considera una práctica recomendada para actualizar usuarios de HTTP a HTTPS. escriba este código en el archivo htaccess para PHP-APACHE

```
Redirect 301 /oldpage/ /newpage/
```

Aquí hay un ejemplo que usa un archivo htaccess para redirigir a un sitio web no www con un SSL adjunto al dominio.

```
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteCond %{HTTP_HOST} ^www\.(.*)$ [NC]
RewriteRule ^(.*)$ http://%1/$1 [R=301,L]

RewriteCond %{HTTPS} on
RewriteCond %{HTTP_HOST} ^www\.(.*)$ [NC]
```

```
RewriteRule ^(.*)$ https://%1/$1 [R=301,L]

RewriteEngine On
RewriteCond %{SERVER_PORT} 80
RewriteRule ^(.*)$ https://example.com/$1 [R,L]
```

Lea Archivos .htaccess en Apache en línea: https://riptutorial.com/es/apache/topic/2089/archivos--htaccess-en-apache

Capítulo 3: Cómo crear un host virtual en Apache

Observaciones

El punto de entrada principal para `VirtualHost` de Apache se encuentra en la documentación del host virtual de Apache . A partir de ahí, tiene documentación general sobre la configuración del host virtual y también documentación de referencia sobre `VirtualHost` y directivas relacionadas.

Examples

Configuración de host virtual basada en nombre

El alojamiento virtual basado en nombre en Apache se describe en el sitio web de Apache como tal:

> Con el alojamiento virtual basado en el nombre, el servidor confía en el cliente para informar el nombre del host como parte de los encabezados HTTP. Usando esta técnica, muchos hosts diferentes pueden compartir la misma dirección IP.

Por lo tanto, más de un sitio web se puede alojar en un servidor a través de este método. En ubuntu, los archivos de configuración están en / etc / apache2 / sites-available. En ese directorio, encontrará 000-default.conf. Esa es la configuración predeterminada, todas las solicitudes se enviarán a este archivo de configuración hasta que se hayan configurado otras.

Para configurar un host virtual, aquí se utilizará **example.com** , pero debe reemplazarlo con su **dominio.com** . Copia el archivo por defecto:

```
cp 000-default.conf example.com.conf
```

El archivo de configuración puede tener las siguientes directivas:

```
<VirtualHost *:80>
        ServerAdmin admin@example.com
        ServerName example.com
        ServerAlias www.example.com

        DocumentRoot /var/www/example.com/html

        ErrorLog /var/log/apache/logs/error.log
        # Possible values include: debug, info, notice, warn, error, crit,
        # alert, emerg.
        LogLevel warn

        CustomLog /var/log/apache/logs/access.log combined
</VirtualHost>
```

- La primera línea indica que todas las solicitudes en el puerto 80 (puerto http predeterminado) deben coincidir. También puede tener una dirección IP en lugar de *, que es la IP del servidor.
- `ServerAdmin` es la información de contacto del administrador del sitio web que se usa para mostrar mensajes de error http.
- `ServerName` es el nombre de dominio del sitio web.
- `ServerAlias` es un nombre secundario del sitio web, generalmente será www.domain.com
- `DocumentRoot` es la carpeta raíz que se carga cuando exploramos un sitio web.
- `ErrorLog` es el archivo donde se dirigen los errores.
- `LogLevel` . es el nivel de errores que se enviarán al registro
- `CustomLog` es el archivo donde se dirige la información de acceso.

Edite el archivo reemplazando example.com con el nombre de dominio de su sitio web y el directorio apropiado para los archivos del sitio web.

Guarde el archivo y habilite el sitio con el siguiente comando de Apache:

```
sudo a2ensite example.com.conf
```

Recargar apache

```
sudo service apache2 reload
```

Algunas cosas más que deben ser revisadas:

- Asegúrese de que su DNS para su dominio esté configurado para la IP correcta (esto puede llevar tiempo para propagarse)
- Asegúrese de que su puerto 80 esté abierto en el firewall
- Asegúrese de que los permisos de archivo estén configurados correctamente en los archivos del servidor: la propiedad debe ser www-data: www-data y los permisos de directorio deben ser 750 y los permisos de archivo deben ser 640.

¡Tu host virtual debería estar en funcionamiento! Puede repetir esto para otros sitios web en el mismo servidor, con un archivo de configuración diferente (utilizando la misma convención de nomenclatura) y diferentes directorios en / var / www.

Desarrollo virtual de PHP Host

Este es un ejemplo sobre cómo controlar el registro de errores de PHP en un sitio de host virtual para el desarrollo y la depuración. Suposiciones

- El módulo de PHP ha sido instalado.
- El entorno de desarrollo no es para la producción.

```
<VirtualHost *:80>
    ServerName example.com
    DocumentRoot /var/www/domains/example.com/html
    ErrorLog /var/www/domains/example.com/apache.error.log
    CustomLog /var/www/domains/example.com/apache.access.log common
```

```
    php_flag log_errors on
    php_flag display_errors on
    php_value error_reporting 2147483647
    php_value error_log /var/www/domains/example.com/php.error.log
</VirtualHost>
```

Nota: la configuración del host virtual es solo para desarrollo porque los display_errors están habilitados y no desea que estén en producción.

Host virtual en WAMP

Suponiendo que está trabajando con Windows 7 PC

Paso 1: GOTO -> C: \ Windows \ System32 \ drivers \ etc Donde encontrará un archivo llamado "hosts", cópielo por favor y péguelo en la misma ubicación. Allí se creará un archivo de copia de los hosts.

Ahora necesitamos hacer algunas modificaciones en este archivo, pero si intenta editarlo con cualquier editor como notepad o notepad ++, no le permitirá guardar el archivo.

Ahora vuelva a copiar el mismo archivo y péguelo en su escritorio, ahora puede editar este archivo fácilmente.

Encontrará una o varias entradas como: 127.0.0.1 localhost en ese archivo. Ahora agregue otra línea debajo de esa línea, por ejemplo: 127.0.0.1 myproject1.local De esta manera, ha definido un nuevo subdominio "myproject1.local" que puede funcionar en lugar de "localhost / myproject1".

Paso 2: Bueno, ahora es el momento de definir la ruta raíz para acceder a este dominio recién creado, ¿no? GOTO: C: \ wamp \ bin \ apache \ Your-Apache-Version \ conf \ extra Aquí encontrará un archivo llamado "httpd-vhosts". Ábrelo en el editor y pega las siguientes líneas en él.

```
<VirtualHost *:80>
    ServerAdmin webmaster@dummy.example.com
    DocumentRoot "c:/wamp/www/myproject1/”
    ServerName myproject1.local
    ErrorLog "logs/myproject1.local-error.log"
    CustomLog "logs/myproject1.local.log" common
</VirtualHost>
```

Ahora ya casi está allí para acceder al proyecto que reside en "c: / wamp / www / myproject1 /"

Paso 3: GOTO: C: \ wamp \ bin \ apache \ your-Apache-Version \ conf

Encuentre un archivo llamado **"httpd.conf"** , cópielo y péguelo en el mismo lugar para su seguridad. Abra el archivo en el editor y busque la palabra **"# hosts virtuales"** , a continuación encontrará la línea **"Incluir conf / extra / httpd-vhosts.conf"** Si está comentada, haga que no haga comentarios y reinicie los servicios de su wamp-server.

Vaya a su navegador web y escriba myproject1.local, puede ver el proyecto en ejecución ahora.

Ahora puede enfrentar un problema que su localhost no funcionará usando localhost como una URL. Sin preocupaciones ... pegue este código en el archivo **"httpd-vhosts"** .

```
<VirtualHost *:80>
    ServerAdmin webmaster@dummy.example.com
    DocumentRoot "c:/wamp/www"
    ServerName localhost
    ErrorLog "logs/localhost-error.log"
    CustomLog "logs/localhost.log" common
</VirtualHost>
```

Reinicie todos los servicios de WAMP, el trabajo está hecho.

Gracias y saludos **Chintan Gor**

1) vhosts basados en IP 2) múltiples vhosts con el mismo puerto 3) definiendo vhosts usando macro (Apache2.4)

1) vhosts basados en IP

```
 <VirtualHost 192.168.13.37>
     ServerName example.com
     DocumentRoot /var/www/domains/example.com/html
     ErrorLog /var/log/example.com/error.log
     CustomLog /var/log/example.com/access.log common
 </VirtualHost>

 <VirtualHost 192.168.47.11>
     ServerName otherurl.com
     DocumentRoot /srv/www/htdocs/otherurl.com/html
     ErrorLog /var/log/otherurl.com/error.log
     CustomLog /var/log/otherurl.com/access.log common
 </VirtualHost>
```

Simplemente cambie el puerto a su IP (s) dada (s). El puerto es irrelevante para la decisión que se elige vhost.

2) múltiples vhosts con el mismo puerto

Ya que NameVirtualHost ya no es necesario, solo puede escribir varios vhosts con el mismo puerto.

```
<VirtualHost *:80>
     DocumentRoot /srv/www/htdocs/otherurl.com/html
     ErrorLog /var/log/otherurl.com/error.log
     CustomLog /var/log/otherurl.com/access.log common
</VirtualHost>

<VirtualHost *:80>
     ServerName example.com
     ServerAlias ex1.com ex2.com
     DocumentRoot /var/www/domains/example.com/html
     ErrorLog /var/log/example.com/error.log
     CustomLog /var/log/example.com/access.log common
```

```
 </VirtualHost>
```

Aquí se aplica lo contrario: la IP es irrelevante, pero si la solicitud se recibe en el puerto 80, se evalúa el nombre que ingresó. ¿Llamaste a ex1.com el segundo vhost es elegido? Y si llamó a cualquier otra url (como otherurl.com, pero también example3.com) se elegirá la primera. Puede utilizar este vhost como un 'respaldo' si lo desea.

3) Definiendo vhosts usando Macro (Apache2.4)

```
<Macro VHost $port $host>
    <VirtualHost *:$port>
        Servername $host
        DocumentRoot /srv/www/htdocs/$host
        ErrorLog /var/log/$host/error.log
    </VirtualHost>
</Macro>

Use VHost 80 example.com
Use VHost 443 secure_example.com
```

Crea dos vhosts, uno para el puerto 80, uno para 443, y establece las variables utilizadas en consecuencia.

Forzar HTTPS usando host virtual

Use **Redirigir** para obligar a los usuarios a conectarse a la URL segura.

```
<VirtualHost *:80>
    ServerName example.com
    SSLProxyEngine on
    Redirect permanent / https://secure_example.com/
</VirtualHost>
```

El resto de la configuración se puede colocar en el host virtual ssl (puerto 443) ya que todo se redirige.

```
<VirtualHost _default_:443>
    ServerName secure_example.com
    ServerAdmin webmaster@example.com
    DocumentRoot /var/www/domains/secure_example.com/html
    ErrorLog /var/log/secure_example.com/error.log
    CustomLog /var/log/secure_example.com/access.log common
    SSLEngine On
    ...
</VirtualHost>
```

Lea Cómo crear un host virtual en Apache en línea:
https://riptutorial.com/es/apache/topic/4856/como-crear-un-host-virtual-en-apache

Capítulo 4: Flujo de apache

Introducción

`Apache Flume` es una herramienta / servicio / mecanismo de ingestión de datos para recopilar y transportar grandes cantidades de datos de transmisión por secuencias, como archivos de registro, eventos (etc.) de varias fuentes a un **almacén de datos centralizado** .

Flume es una herramienta altamente confiable, distribuida y configurable. Está diseñado principalmente para copiar datos de transmisión (datos de registro) de varios servidores web a `HDFS` .

Examples

Transmisión / registro de datos

En general, la mayoría de los datos que se analizarán serán producidos por varias fuentes de datos, como servidores de aplicaciones, sitios de redes sociales, servidores en la nube y servidores empresariales. Estos datos estarán en forma de archivos de registro y eventos.

Archivo de registro: en general, un archivo de registro es un archivo que enumera eventos / acciones que ocurren en un sistema operativo. Por ejemplo, los servidores web enumeran todas las solicitudes realizadas al servidor en los archivos de registro.

En la recolección de dichos datos de registro, podemos obtener información sobre:

El rendimiento de la aplicación y localizar diversos fallos de software y hardware. El comportamiento del usuario y obtener mejores perspectivas de negocio. El método tradicional de transferir datos al sistema HDFS es usar el comando put. Veamos cómo usar el comando put.

Lea Flujo de apache en línea: https://riptutorial.com/es/apache/topic/9630/flujo-de-apache

Creditos

S. No	Capítulos	Contributors
1	Empezando con apache	Community, fab, Flamewires, hjpotter92, James, Katie, Kuhan, Nicholas Qiao, Rich Bowen
2	Archivos .htaccess en Apache	Chintan Gor, Deltik, ezra-s, Luke Bearl, Rich Bowen, SimpleAnecdote
3	Cómo crear un host virtual en Apache	Chintan Gor, Clutch, fab, Harikrishnan, Hello Fishy, Katie, Olaf Dietsche
4	Flujo de apache	Vinod Kumar