EBook Gratuito

APPRENDIMENTO

apache

Free unaffiliated eBook created from
Stack Overflow contributors.

#apache

Sommario

Di

Capitolo 1: Iniziare con Apache

Osservazioni

Questa sezione fornisce una panoramica di ciò che è Apache e perché uno sviluppatore potrebbe voler usarlo.

Dovrebbe anche menzionare tutti i soggetti di grandi dimensioni all'interno di apache e collegarsi agli argomenti correlati. Poiché la documentazione di apache è nuova, potrebbe essere necessario creare versioni iniziali di tali argomenti correlati.

Versioni

Varie versioni di httpd di Apache

Versione	Versione corrente	pubblicazione
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

Installazione o configurazione

Istruzioni dettagliate su come installare o installare Apache.

Installazione di Ubuntu

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

Installazione di Windows

Controlla lo stack WAMP . WAMP sta per Windows, Apache, MySQL, PhpMyAdmin.

Installazione di CentOS

Apache 2.2 viene fornito con CentOS6, mentre 2.4 viene fornito con CentOS7, da installare su entrambi i sistemi operativi

```
yum -y install httpd
```

installazione di macOS

macOS viene fornito con Apache preinstallato, tuttavia, può installare Apache tramite Homebrew

Se hai già l'Apache integrato in esecuzione, è necessario prima arrestarlo e rimuovere qualsiasi script di caricamento automatico.

```
$ 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] Esempio Hello Hello World

Questo esempio ti guiderà attraverso l'impostazione di un back-end che serve una pagina HTML di Hello World.

Requisiti di installazione

Ordinare è importante per questo passaggio!

- `sudo apt-get install apache2`

Configurare l'HTML

I file Apache si trovano in `/var/www/html/` . Consente di arrivare rapidamente. Assicurati di essere nella tua directory principale prima, `cd` , quindi `cd /var/www/html/` .

Questa directory `html` è dove vivranno tutti i file del tuo sito web. Consente di creare rapidamente un semplice file Hello World.

Usando il tuo editor di testo preferito, digita quanto segue in

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

Salva questo file come `index.html` nella directory corrente e sei pronto per partire!

Visitando la tua pagina web

Per visitare la pagina che hai appena creato, nel tuo browser di scelta, vai a `localhost` . Se ciò non funziona, prova `127.0.0.1` . Dovresti vedere "Hello World!" come un `h1` . Hai finito!

Per garantire che il server sia attivo.

Se ricevi un messaggio che il browser non è in grado di connettersi al server, per prima cosa controlla che il server sia attivo.

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

Dovresti vedere alcuni processi httpd se Apache è attivo e funzionante.

Leggi Iniziare con Apache online: https://riptutorial.com/it/apache/topic/964/iniziare-con-apache

Capitolo 2: .htaccess file in Apache

Examples

Riscrivi il motore

Il modulo RewriteEngine in Apache viene utilizzato per riscrivere dinamicamente URL e percorsi in base alle varie espressioni fornite:

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

Le regole di cui sopra riscriveranno i file PHP per non mostrare più la loro estensione, e così index.php mostrerà solo un dominio nudo (simile al comportamento normalmente visto in index.html). La regola precedente viene fornita con WordPress.

Notare che in Apache httpd 2.2.16 e successive, questo intero blocco può essere sostituito con una singola riga usando la direttiva FallbackResource:

```
FallbackResource /index.php
```

Forza HTTPS

.htaccess può essere usato per forzare il tuo sito HTTP a reindirizzare a HTTPS.

Ecco un modo rapido che non richiede la modifica del codice per il tuo dominio:

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

> **Avviso.** Il codice precedente presuppone che puoi fidarti di `%{HTTP_HOST}` in modo che punti al tuo dominio.
>
> Se devi essere sicuro che la posizione di reindirizzamento è il tuo dominio, sostituisci `%{HTTP_HOST}` con il tuo dominio.

Il codice sopra fa questo:

1. Abilita RewriteEngine .
2. Continua se la richiesta corrente non sta utilizzando HTTPS.
3. `https://%{HTTP_HOST}%{REQUEST_URI}` un reindirizzamento HTTP 301 a

`https://%{HTTP_HOST}%{REQUEST_URI}` , dove

- `%{HTTP_HOST}` è l'host richiesto dal browser e
- `%{REQUEST_URI}` è l'URI richiesto dal browser (tutto dopo il dominio).

Avviso: l' applicazione Web deve essere in grado di gestire le richieste HTTPS e Apache per l'host deve essere configurato con un certificato del sito valido.

Si noti che è significativamente più efficiente eseguire semplicemente un `Redirect` nel http vhost piuttosto che effettuare questi confronti multipli per richiesta in un file .htaccess. Vedi http://wiki.apache.org/httpd/RedirectSSL per ulteriori discussioni su questa tecnica.

Abilita CORS

Per abilitare ***Cross-Origin Resource Sharing*** (**CORS**) in Apache è necessario impostare almeno un'intestazione HTTP che la modifichi (il comportamento predefinito è bloccare CORS). Nell'esempio seguente, imposteremo questa intestazione HTTP all'interno di `.htaccess` , ma può anche essere impostata nel tuo sito il `your-site.conf` file `your-site.conf` o il file di configurazione di Apache. Indipendentemente dal tipo di configurazione, è possibile impostare le intestazioni HTTP pertinenti in qualsiasi blocco di configurazione Apache, ovvero `<VirtualHost>` , `<Directory>` , `<Location>` e `<Files>` .

Ci sono alcune intestazioni HTTP relative a CORS che puoi restituire nella risposta:

```
Access-Control-Allow-Origin

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

Alcuni di questi sono richiesti per le richieste di "preflight". Alcuni client HTTP (ovvero i browser moderni) eseguono una richiesta **prima** della richiesta desiderata solo per verificare se dispongono dell'autorizzazione per effettuare la richiesta effettiva sul server. Vedi https://en.wikipedia.org/wiki/Cross-origin_resource_sharing per ulteriori informazioni sulla richiesta di preflight.

L'intestazione HTTP principale di cui abbiamo bisogno è `Access-Control-Allow-Origin` e stiamo andando a impostare. Tuttavia, lo stesso principio si applica praticamente a tutti (devi solo sapere cosa restituire).

L'esempio seguente imposta l'intestazione HTTP richiesta all'interno di un blocco di configurazione `<Directory>` per abilitare un nome dominio completo (FQDN) completo protetto da SSL:

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

Dopo averlo impostato sul **server** , ora possiamo eseguire una richiesta da https:

//my.client.domain al nostro server e dovrebbe rispondere.

Nota: molte persone usano `Access-Control-Allow-Origin: "*"` che è un carattere jolly, per indicare che le richieste provenienti da **TUTTI i** domini devono essere accettate. Questo di solito è sconsigliato a meno che tu non stia utilizzando una sorta di API pubblica o repository di file. Inoltre, tieni presente il contesto della tua impostazione dell'intestazione HTTP. Potresti voler consentire le richieste HTTP per un'API, ma non per le immagini di "hotlinking" ecc. Puoi impostare questa intestazione ovunque nel flusso di configurazione di Apache per impostarla **solo** in situazioni specifiche. Ad esempio, il seguente comando imposta l'intestazione HTTP CORS **solo** quando il percorso richiesto **non** è un file o una directory (si adatta a un'API pubblica che non consente l'hotlinking dell'immagine):

```
<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>
```

Prerequisiti

Devi avere mod_headers installati e abilitati: `a2enmod headers`

301 Reindirizzamento di Htaccess

Il codice di stato della risposta HTTP 301 Spostato in modo permanente viene utilizzato per il reindirizzamento permanente dell'URL, ovvero i collegamenti o i record correnti che utilizzano l'URL per il quale la risposta viene ricevuta devono essere aggiornati. Il nuovo URL deve essere fornito nel campo Posizione incluso nella risposta. Il reindirizzamento 301 è considerato una best practice per l'aggiornamento degli utenti da HTTP a HTTPS. scrivi questo codice nel file htaccess per PHP-APACHE

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

Ecco un esempio che utilizza un file htaccess per reindirizzare a un non www con un SSL collegato 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]
```

Leggi .htaccess file in Apache online: https://riptutorial.com/it/apache/topic/2089/-htaccess-file-in-apache

Capitolo 3: Apache Flume

introduzione

`Apache Flume` è un meccanismo di acquisizione di strumenti / servizi / dati per la raccolta di aggregazione e trasporto di grandi quantità di dati di streaming come file di registro, eventi (ecc.) Da varie fonti a un **archivio dati centralizzato** .

Flume è uno strumento altamente affidabile, distribuito e configurabile. È progettato principalmente per copiare i dati di streaming (dati di registro) da vari server Web su `HDFS` .

Examples

Streaming / Dati di registro

Generalmente, la maggior parte dei dati che devono essere analizzati saranno prodotti da varie fonti di dati come server di applicazioni, siti di social networking, server cloud e server aziendali. Questi dati saranno sotto forma di file di registro ed eventi.

File di registro: in generale, un file di registro è un file che elenca eventi / azioni che si verificano in un sistema operativo. Ad esempio, i server Web elencano ogni richiesta effettuata al server nei file di registro.

Al momento della raccolta di tali dati di registro, possiamo ottenere informazioni su -

le prestazioni dell'applicazione e individuare vari errori software e hardware. il comportamento degli utenti e derivano migliori intuizioni di business. Il metodo tradizionale di trasferimento dei dati nel sistema HDFS consiste nell'utilizzare il comando put. Vediamo come usare il comando put.

Leggi Apache Flume online: https://riptutorial.com/it/apache/topic/9630/apache-flume

Capitolo 4: Come creare un host virtuale in Apache

Osservazioni

Il punto di ingresso principale per `VirtualHost` di Apache è nella documentazione di Apache Virtual Host . Da lì, si dispone di documentazione generale sulla configurazione dell'host virtuale e della documentazione di riferimento su `VirtualHost` e le relative direttive.

Examples

Configurazione dell'host virtuale basata su nome

L'hosting virtuale basato su nome su Apache è descritto nel sito Web di Apache in quanto tale:

> Con l'hosting virtuale basato sul nome, il server si affida al client per segnalare il nome host come parte delle intestazioni HTTP. Usando questa tecnica, molti host diversi possono condividere lo stesso indirizzo IP.

Pertanto, più di un sito Web può essere ospitato su un server tramite questo metodo. Su ubuntu, i file di configurazione sono in / etc / apache2 / sites-available. In quella directory, troverai 000-default.conf. Questa è la configurazione di default, tutte le richieste verranno inviate a questo file di configurazione fino a quando altre non saranno state configurate.

Per configurare un host virtuale, verrà utilizzato qui **example.com** , ma dovresti sostituirlo con il tuo **dominio.com** . Copia il file predefinito:

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

Il file di configurazione può avere le seguenti direttive:

```
<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 prima riga indica che tutte le richieste sulla porta 80 (porta HTTP predefinita) devono

corrispondere. Puoi anche avere un indirizzo IP invece di * che è l'IP del server.
- `ServerAdmin` è i dettagli di contatto dell'amministratore del sito Web utilizzato per la visualizzazione con messaggi di errore http.
- `ServerName` è il nome di dominio del sito Web.
- `ServerAlias` è un nome secondario del sito Web, di solito sarà www.domain.com
- `DocumentRoot` è la cartella principale caricata quando navighiamo un sito web.
- `ErrorLog` è il file in cui sono diretti gli errori
- `LogLevel` . è il livello di errori da inviare al log
- `CustomLog` è il file in cui sono indirizzate le informazioni di accesso

Modifica il file che sostituisce example.com con il nome del dominio del tuo sito web e la directory appropriata per i file del sito web.

Salvare il file e abilitare il sito con il seguente comando Apache:

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

Ricarica apache

```
sudo service apache2 reload
```

Alcune altre cose che devono essere verificate:

- Assicurati che il tuo DNS per il tuo dominio sia impostato per l'IP corretto (potrebbe essere necessario del tempo per propogare)
- Assicurarsi che la porta 80 sia aperta sul firewall
- Assicurati che le autorizzazioni dei file siano configurate correttamente sui file del server: la proprietà dovrebbe essere www-data: i dati di www-data e le autorizzazioni di directory dovrebbero essere 750 e le autorizzazioni di file dovrebbero essere 640.

Il tuo host virtuale dovrebbe essere attivo e funzionante! È possibile ripetere questo per altri siti Web sullo stesso server, con un file di configurazione diverso (utilizzando la stessa convenzione di denominazione) e diverse directory in / var / www.

Host virtuale di sviluppo PHP

Questo è un esempio su come controllare la registrazione degli errori PHP in un sito host virtuale per lo sviluppo e il debug. ipotesi

- Il modulo PHP è stato installato.
- L'ambiente di sviluppo non è per la produzione.

```
<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 configurazione dell'host virtuale è solo per lo sviluppo perché display_errors è abilitato e non lo si desidera in produzione.

Host virtuale in WAMP

Supponendo che tu stia lavorando con Windows 7 PC

Passo 1: GOTO -> C: \ Windows \ System32 \ drivers \ etc Dove troverai un file chiamato "hosts", copialo gentilmente e incollalo nella stessa posizione. Lì verrà creato un file di copia degli host.

Ora dobbiamo fare alcune modifiche in questo file, ma se provi a modificarlo con qualsiasi editor come notepad o notepad ++, non ti permetterà di salvare il file.

Ora copia nuovamente lo stesso file e incollalo sul desktop, ora puoi modificare facilmente questo file.

Troverai una o più voci come: 127.0.0.1 localhost in quel file. Ora aggiungi un'altra linea sotto quella linea, ad esempio: 127.0.0.1 myproject1.local In questo modo hai definito un nuovo sottodominio "myproject1.local" che può funzionare al posto di "localhost / myproject1".

Passo 2: Ok, ora è il momento di definire il percorso root per accedere a questo dominio appena creato giusto? GOTO: C: \ wamp \ bin \ apache \ Your-Apache-Version \ conf \ extra Qui troverai un file chiamato "httpd-vhosts". Aprilo nell'editor e incolla le righe sottostanti in esso.

```
<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>
```

Ora sei quasi lì per accedere al progetto che risiede in "c: / wamp / www / myproject1 /"

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

Trova un file chiamato **"httpd.conf"** , copialo e incollalo nello stesso posto per sicurezza. Apri il file nell'editor e trova una parola **"# host virtuali"** , di seguito troverai una riga **"Includi conf / extra / httpd-vhosts.conf"** Se viene commentata, **rendila** non commentata e riavvia i servizi del server wamp.

Vai al tuo browser web e scrivi myproject1.local, puoi vedere ora il progetto in esecuzione.

Ora potresti affrontare un problema che il tuo localhost non funzionerà usando localhost come URL. No Worries ... incolla questo codice nel file **"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>
```

Riavvia tutti i servizi di WAMP, il lavoro è fatto.

Grazie e **saluta Chintan Gor**

1) Vhosts basati su IP 2) Multiple vhosts con la stessa porta 3) Definizione di vhosts usando Macro (Apache2.4)

1) vhosts basati su 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>
```

Basta cambiare la porta per il tuo IP (s) dato. La porta è irrilevante per la decisione su quale vhost è stato scelto.

2) Vhosts multipli con la stessa porta

Dal momento che NameVirtualHost non è più necessario, puoi semplicemente scrivere più vhosts con la stessa porta.

```
<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>
```

Qui vale il contrario: l'IP è irrilevante, ma se la richiesta viene ricevuta sulla porta 80 viene valutato

il nome inserito. Hai chiamato ex1.com e il secondo vhost viene selezionato. E se hai chiamato un altro URL (come otherurl.com, ma anche example3.com) verrà selezionato il primo. Puoi usare questo vhost come "riserva" se vuoi.

3) Definizione di 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 due vhosts, uno per la porta 80, uno per 443 e imposta di conseguenza le variabili utilizzate.

Forza HTTPS utilizzando l'host virtuale

Utilizza **Reindirizzamento** per costringere gli utenti a connettersi all'URL protetto.

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

Il resto della configurazione può essere inserito nell'host virtuale ssl (porta 443) poiché tutto viene reindirizzato.

```
<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>
```

Leggi Come creare un host virtuale in Apache online:
https://riptutorial.com/it/apache/topic/4856/come-creare-un-host-virtuale-in-apache

Titoli di coda

S. No	Capitoli	Contributors
1	Iniziare con Apache	Community, fab, Flamewires, hjpotter92, James, Katie, Kuhan, Nicholas Qiao, Rich Bowen
2	.htaccess file in Apache	Chintan Gor, Deltik, ezra-s, Luke Bearl, Rich Bowen, SimpleAnecdote
3	Apache Flume	Vinod Kumar
4	Come creare un host virtuale in Apache	Chintan Gor, Clutch, fab, Harikrishnan, Hello Fishy, Katie, Olaf Dietsche