



# C1- Code & Go

---

C-WEB-150

# Administration System

---

Apache



# Foreword

---

This first day marks your first step as a System Administrator. You will discover how to manage the server running your various projects.

You must do this project alone, of course we don't mind that you discuss with your peers from time to time.

There is nothing to submit in your repository for today.



Tip: since there is nothing to submit, you are strongly advised to spend this day with your assistants so that you can ask questions at the right time.



# Apt-Get

---

apt-get

Advanced Packaging Tool - APT - is a package manager used by Debian GNU/Linux and its derivatives. It's a command line tool helpful to install easily a package by name.

```
Terminal
~/C-WEB-150> #Useful commands
~/C-WEB-150> apt install <package>
~/C-WEB-150> apt search <words>
~/C-WEB-150> apt remove <package>
~/C-WEB-150> apt update
~/C-WEB-150> apt upgrade
```

When you are doing an apt operation, take the habit to do apt update and apt upgrade before. This ensures that you will always have the last available packages, and the last version of that tool.



# Apache

---



# Apache

---

## Why Apache?

Apache is the most common HTTP server. It is designed to accept many modules which give it additional functionalities (Perl, PHP, Python, Ruby, CGI, SSI, URL rewriting, content negotiation, etc.).

---

## Apache Installation

Type the following to proceed with the installation process

```
Terminal
~/C-WEB-150> #Install Apache
~/C-WEB-150> sudo apt-get install apache2 apache2-doc
```

---

## Configuration

All Apache configuration files are store in Apache **/etc/apache2/** :

- 1. **apache2.conf**: General configuration (formerly httpd.conf)
- 2. **mods-available/**: Available modules
- 3. **mods-enabled/**: Enabled modules
- 4. **sites-available/**: Available sites (contains default for the site hosted by default)
- 5. **sites-enabled/**: Enabled sites

In order to enable the modules, use **a2enmod** (Apache 2 Enable Module) followed by the name of the module. We enable here **mod\_rewrite** for rewriting URLs.

```
Terminal
~/C-WEB-150> #Configure Apache
~/C-WEB-150> sudo a2enmod rewrite
```

In order to activate a site with its configuration stored in sites-available, use **a2ensite** (Apache 2 Enable Site):

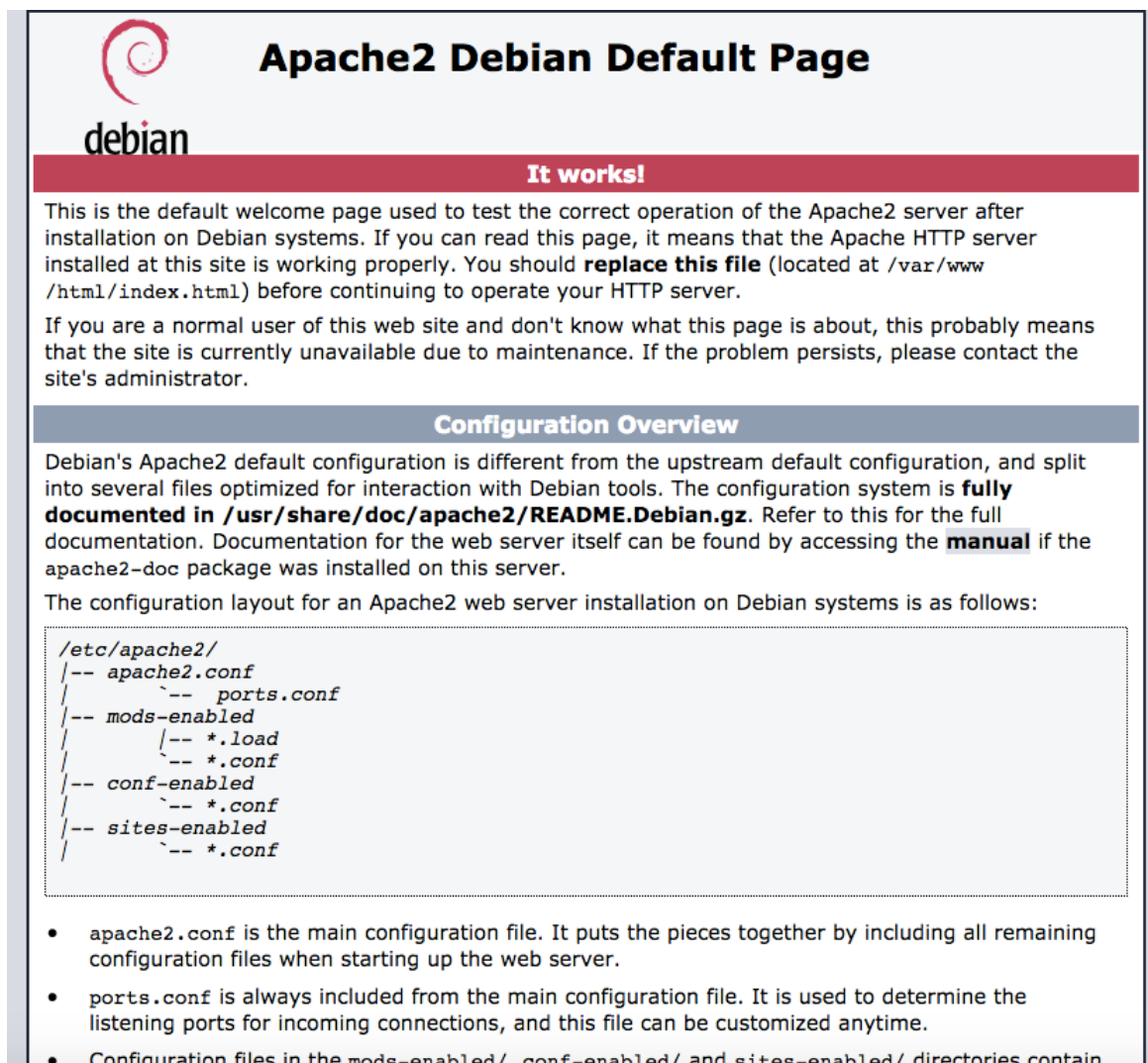


```
Terminal
~/C-WEB-150> #Configure Apache
~/C-WEB-150> sudo a2ensite <siteName>
```

To disable a site, use a2dissite. To disable a module, use a2dismod.  
**Note:** do not modify the content of mods-enabled and sites-enabled!

## Test

Open your browser, and enter localhost in URL. You'll see this page:



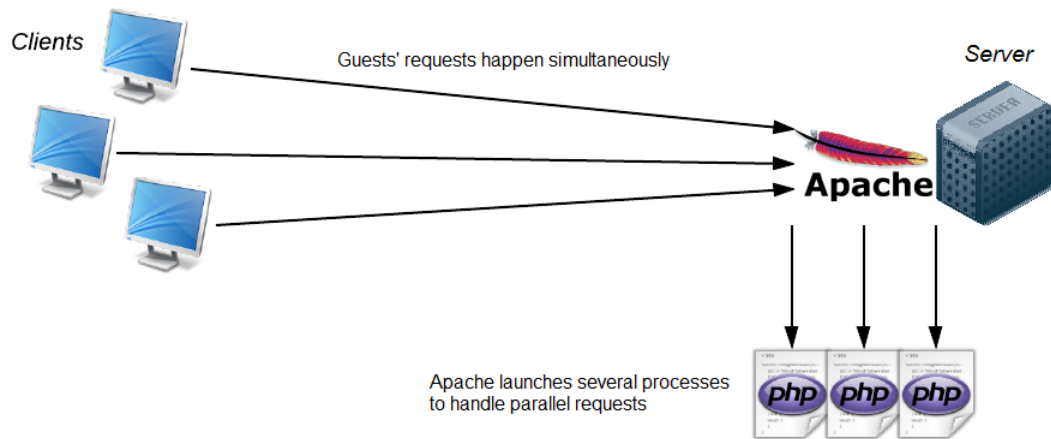
The screenshot shows the Apache2 Debian Default Page. At the top, there is a red swirl logo and the word "debian" in a stylized font. The main heading is "Apache2 Debian Default Page". Below this, a red banner says "It works!". The text explains that this is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. It mentions that if you can read this page, it means the Apache HTTP server is working properly. It also states that you should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server. A section titled "Configuration Overview" follows, explaining that Debian's Apache2 default configuration is different from the upstream default configuration and is split into several files optimized for interaction with Debian tools. It refers to the full documentation in /usr/share/doc/apache2/README.Debian.gz and mentions that documentation for the web server itself can be found by accessing the manual if the apache2-doc package was installed. It then states that the configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

- apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- ports.conf is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/ directories contain



## What happens?





# Virtual Host

---

## Definition

---

Well, with the common configuration, your web site folder is located in:

```
/var/www/html/
```

When you want to access your web site on your browser, type **localhost[:port]**. This is the common local server name. The initial one is pointing at this directory. Unfortunately, this directory should be owned by the super user, so you can't modify the files with your common user.

It's not very friendly or secure to invoke the super user to develop your website, so we'll create a virtual host.

By doing this, we are telling Apache there is another directory acting as a web server. We can name it and give it some particular configuration.

For example, we want to use our entire "Rendu" folder working as a web server. It will be very easy, each of your projects will be accessed in the same way, with no files to copy and no super user to invoke. After doing that, you'll access your "Rendu" folder by **typing coding-academy[:port]** on your browser.



## Configuration

So, go to your apache configuration folder:

```
Terminal
~/C-WEB-150> cd /etc/apache2/sites-available
```

Then, we will copy the default Apache configuration usually named "000-default.conf" and create a new configuration.

**Please, do not move it or change it, this file is very useful to write virtualhost configuration**

```
Terminal
~/C-WEB-150> cp 000-default.conf 000-coding-academy.conf
```

Now, we'll modify our new coding-academy configuration:

```
Terminal
~/C-WEB-150> emacs 000-codig-academy.conf
```

Please make sure it looks like the following

```
Terminal
~/C-WEB-150> cat 000-default.conf
ServerAdmin you@coding-academy.com
ServerName coding-academy.com

ServerAlias www.coding-academy.com
DocumentRoot /path/to/your/folder
ErrorLog ${APACHE_LOG_DIR}/coding-academy.error.log
CustomLog ${APACHE_LOG_DIR}/coding-academy.access.log combined
```

One more configuration file and we're done, and this is the main apache configuration file.

**/etc/apache2/apache2.conf**

In this file, you will find some configuration looking like this:

```
Terminal
~/C-WEB-150> cat /etc/apache2/apache2.conf
<Directory /var/www/>
    Options FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

You just have to copy it and replace the "Directory" rule (ie : /var/www) with your folder (for example: "/home-/johndoe/Rendu").

You can add one keyword on the "Options" line: Indexes. It's very useful: with it, you can list the files on the current directory if no index files were found.





You have know to tell your OS that "coding-academy.com" is now a valid url and we own it. Edit your hosts configuration in /etc/hosts and add the following line:

```
127.0.0.1 coding-academy.com
```

```
127.0.0.1 www.coding-academy.com
```

To enable your new configuration please run:

```
Terminal
~/C-WEB-150> sudo systemctl restart apache2
~/C-WEB-150> sudo a2ensite 000-coding-academy
```

The preceding command may not work, but try to access it from your browser as explained in the following.

**You now can access your directory by hitting coding-academy.com in your browser. If you can't access your web page, please verify that the permissions of the whole path to your file is correct.**



# Apache Security

---



## Apache

We are now going to see how to set minimum security to your previously installed web server.  
An Apache server with basic configuration makes some information available that can be used by sketchy people  
We are now going to deal with this

---

### Server information

When your server encounters an error, and cannot process the request, Apache will by default, give information about the server's type and version.

In your browser, try to access a page that does not exist on your server and you will see what we mean:

`http://localhost/doesn't_exists`

- 1. Go to folder: `etc/apache2`
- 2. Edit the file: `conf-available/security.conf`
- 3. Locate the line: `ServerTokens`, and change its value to: `Prod`
- 4. Locate the line: `ServerSignature`, and change its value to: `Off`
- 5. Restart apache

```
Terminal
~/C-WEB-150> sudo systemctl restart apache2
```

Refresh the page that does not exist.



## Disable folder's content listing

By default, when you access a folder of your server without any index.html or index.php file, Apache will list the content of the folder and enable its file tree.

To disable this behavior, open the file `apache2.conf` in `/etc/apache2`

The following actions are only for tests and understanding. You must undo what you will do for this part to continue. Then modify the directive `Options` of `Directory /var/www` like this:

```
Terminal
~/C-WEB-150> nano /etc/apache2/apache2.conf #or with any editor of your choice
<Directory /var/www/>
    Options FollowSymLinks Multiviews
    AllowOverride None
    Require all granted
    deny from all
    allow from 127.0.0.1
</Directory>
```

Do that for `/home/<your_login>/Rendu`:

```
Terminal
~/C-WEB-150> nano /etc/apache2/apache2.conf #or with any editor of your choice
<Directory /home/<your_login>/Rendu>
    Options FollowSymLinks Multiviews
    AllowOverride None
    Require all granted
    deny from all
    allow from 127.0.0.1
</Directory>
```

Finally, restart Apache to apply changes.

## IP Filtering

If you have a site which can only be accessed by a certain network or IP address, you can filter the authorized IPs by editing the `apache2.conf`.

These are examples, the three together don't really mean anything.

```
Terminal
~/C-WEB-150> nano /etc/apache2/apache2.conf #or with any editor of your choice
#The client must correspond to the ip
    Require ip 176.16.1.1
#Accept all clients
    Require all granted
#Refuse all requests
    Require all denied
```

Or by using some "Require" scopes:



```
Terminal
~/C-WEB-150> nano /etc/apache2 #or with any editor of your choice
<RequireAny>
    #require one of the next rule to be true
    <RequireAll>
        #require all the next rules to be true
        <RequireNone>
            #require all the next rules to be false
            Require host bad.host.com
        </RequireNone>
    </RequireAll>
    <RequireAll>
        Require user www-data
        Require ip 127.0.0.1
    </RequireAll>
</RequireAll>
<RequireAll>
    #require all the next rules to be true
    Require user root
    Require ip 123.123.123.123
</RequireAll>
</RequireAny>
```

Play a little bit with those rules.

To understand how to use them, take a quick (or long, it depend how brave you are) look at

<http://httpd.apache.org/docs/2.4/en/howto/access.html>

to get detailed information.



# .htaccess

---



---

## Definition

The .htaccess are configuration files of HTTP Apache servers. They are commonly used to configure access rights, url redirects, customized error messages and other useful things for the configuration.

Be careful, it's totally normal if you cannot see a .htaccess file with a simple 'ls -l' on your terminal. Think about the meaning of '.'

---

## Password protection

To protect a folder with a password, you only need to create a .htaccess file and a .htpasswd file in the folder in question.

The directive **AllowOverride** must be set to **all** for the directory in question. (You should now know which file you must modify, don't forget to restart Apache after modification). Inside your prod folder, create a .htaccess file with the following content:

```
Terminal
~/C-WEB-150> nano .htaccess
AuthName "Secure Zone"
AuthType Basic
AuthUserFile "/var/www/html/prod/.htpasswd"
Require valid-user
```

Go to <http://www.htaccesstools.com/htpasswd-generator/> and generate the directive for the .htpasswd by entering the user-pass you want.

Copy the result in the .htpasswd file, and allow the user named teacher to connect (the password is teacher too, but encrypted below).

```
teacher:$apr1$uuhClQqC$cyZSwYrtWVYAynRzrqvpi/
```



## Possible actions

- 1. Prevent the listing of a directory content To prevent users from listing all the files contained in a directory when there is no index (.html, .php, etc.), create a **.htaccess** file containing the line below:

```
Options -Indexes
```

- 2. Redirecting error messages If you want to use customized error messages or redirect the errors of a web page, create or edit a **.htaccess** file containing lines of this type:

```
ErrorDocument error_number message_or_destination
```

The 3 most common errors are:

- 1. 404: not found,
- 2. 403: forbidden,
- 3. 500: internal server error.

Replace "error\_number" by the corresponding number and replace "message\_or\_destination" by the action to perform. To display a simple message, type in the corresponding message between quotes. To redirect to a page, write the path to the page. Here are 2 examples that can help you:

- 1. You wish to display "Sorry, you cannot access this file" in case of a 403 error. Add the following line to your .htaccess:

```
ErrorDocument 403 "Sorry, you cannot access this file"
```

- 2. You wish to send 404 errors on your customized page 404.html:

```
ErrorDocument 404 404.html
```

- 3. Specify a different index file By default, the index file of a directory is index.html, index.htm or index.php. If you want another file, you can add a line of this type to your .htaccess:

```
DirectoryIndex name_of_file
```

For example, if you want to use the page home.html as index page, use the following line:

```
DirectoryIndex home.html
```



---

## Permanent redirect

This sends a permanent redirect HTTP 301 status code which informs users, and more importantly search engines, that they must update their links with the new address. (if you don't know what 301 or HTTP code mean, say hi to Google and its wonderful search bar)

- 1. Redirect the whole site to a new address

```
Redirect 301 / http://new-site.tld/
```

- 2. Redirect a directory or a file to a new one

```
Redirect 301 /old_directory http://new-site.tld/new_directory
```

```
Redirect 301 /old_file.php http://new-site.tld/new_file.php
```

---

## Rewriting url

We can rewrite url with .htaccess and the mod\_rewrite module of Apache. Fantastic, now you know that Apache has modules to enable or disable!

This formula redirects each request on the script testing.php:

```
RewriteEngine On
```

```
RewriteRule .* testing.php
```

Try this one, can you understand what it does?

```
RewriteEngine On
```

```
RewriteRule letstest /test_wslash/testing.php
```

This chapter is important and deep, be curious and check the online documentation of Apache about mod\_rewrite and rewrite rules:

[http://httpd.apache.org/docs/current/mod/mod\\_rewrite.html#rewriterule](http://httpd.apache.org/docs/current/mod/mod_rewrite.html#rewriterule)



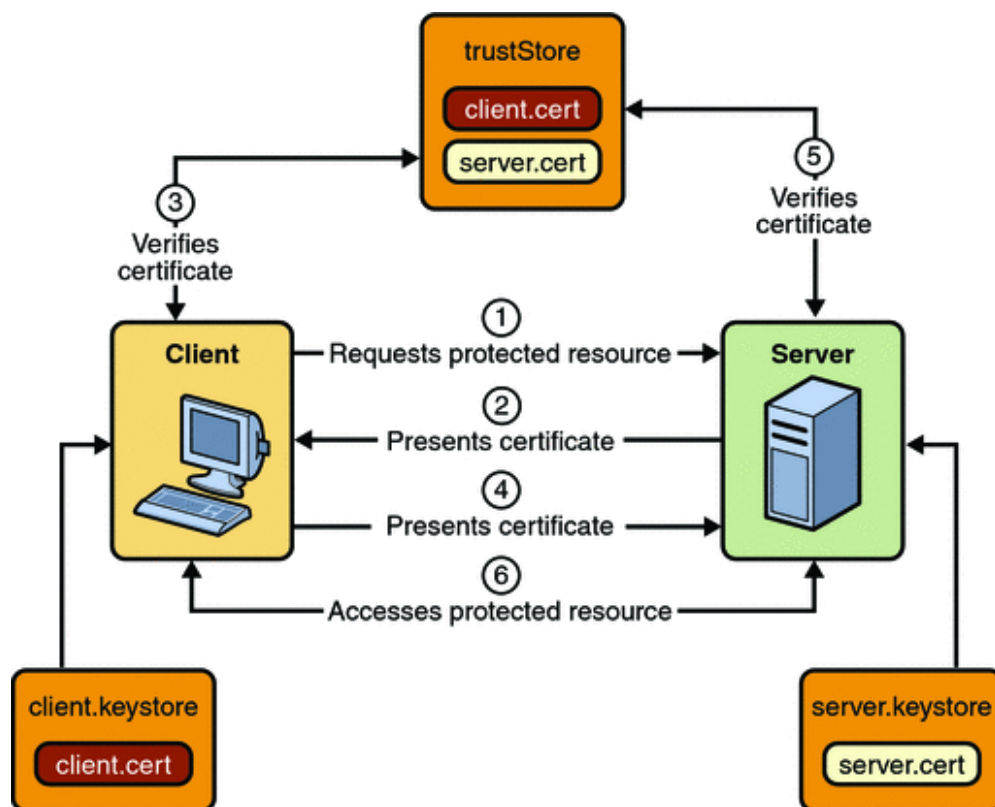
# SSL



## Definition

SSL/TLS (Secure Sockets Layer) is the standard security technology for establishing an encrypted link between a web server and a browser. This link ensures that all data passed between the web server and browsers remain private and complete. Basically, when you see "https" at the beginning of an url, it's a secure website using SSL.

## How it works?







# Load Balancing

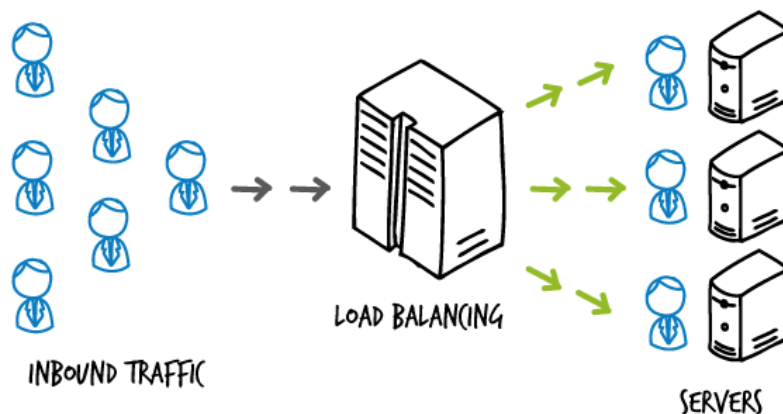
## Definition

Load balancing is a set of techniques which can distribute a workload among various computers of a group. These techniques can both manage service overloads by distributing them on several servers, and reduce eventual unavailability of the service in case of failure of a software or device of a single server 1,2.

These techniques are commonly used, for example, in HTTP services where a high-traffic website must manage hundreds of thousands of requests every second.

The most common architecture is composed of several load balancers (type of routers dedicated to this task), one being the main, and one or several others as backup which can take over, and a collection of similar computers doing a server cluster. The expression server pool is also used. the calculations. We can call this set of servers a server farm or, more generally,

## How it works?



Learn more about load balancing here:

<https://www.nginx.com/resources/glossary/load-balancing/>