

# Chapter 2. Setting up a Web Server

## Table of Contents

2.1	Wampserver and Apache HTTP on Windows.....	1
2.1.1	Requirements.....	1
2.1.2	Installing Wampserver.....	2
2.1.3	Setting up Server Passwords .....	3
2.1.4	Testing Applications.....	5
2.2	Apache Tomcat on Windows .....	6
2.2.1	Requirements.....	6
2.2.2	Tomcat Setup.....	7
2.2.3	Testing Applications.....	10
2.3	Lampserver and Apache HTTP on Ubuntu 15.04 .....	10
2.3.1	Requirements.....	10
2.3.2	Installing Apache, PHP and MySQL.....	11
2.3.3	Testing Applications.....	13
2.4	Apache Tomcat on Ubuntu 15.04.....	13
2.4.1	Testing Applications.....	14

## Objectives

At the end of this unit you will be able to:

- install and setup Wamp server and Apache server on Windows;
- install and setup Tomcat Server on Windows;
- install and setup Lamp server and Apache server on Ubuntu 15.04; and
- install and setup Tomcat on Ubuntu 15.04.

## 2.1 Wampserver and Apache HTTP on Windows

In this section, you will learn how to set up a Web Server on a Windows PC. The steps in this section will illustrate how to use Apache HTTP. The next section will illustrate the setup for Apache Tomcat. Apache is a popular Web Server that allows users to easily set up their own Web Servers. It has the advantage of being open-source and hence is free to download. Apache is the basic software needed to support running of HTML and related content. Additional software, such as Tomcat, can be installed to complement the Web Server. Tomcat is a server that is meant to run applications written in Java and JSP (Java Server Pages).

Some popular options for deploying Apache, and optionally PHP and MySQL on Windows are Apache Lounge, XAMPP and Wampserver. Wampserver was used for this example. WAMP is an acronym that stands for “Windows, Apache, MySQL, and PHP”.

### 2.1.1 Requirements

To illustrate the steps below a Windows 7 64-bit computer was used. The Windows computer was connected to a local area network (LAN) that has Internet access. You also need to know the IP address of your

# Web Servers

computer. You can find your IP address by typing 'ipconfig' at a command prompt. Find the entry labeled 'Ethernet adapter Local Area Network' and take note of the IPv4 address.

It is recommended to disable the Windows Firewall before starting the Web Server setup. The steps below are for a fresh installation of Wampserver (assumes that Wampserver had not been installed before).

## 2.1.2 Installing Wampserver

Download WAMP from <http://www.wampserver.com/en/> . You will have the option of choosing 64-bit or 32-bit installation depending on your PC. This example uses the 64-bit installation. Locate the downloaded Wampserver file and click on it. This will open an installation wizard as shown in Figure 1. Follow the instruction wizard and leave the default settings as they are. After successful installation you will get the window shown in Figure 2. Leave the 'launch WampServer 2 now' box checked and click on 'Finish' button (in future you can start WampServer by clicking on your Start menu and clicking on its menu). On your toolbar, you should now see a 'W' shaped icon. On left-clicking this icon, you get the pop-up management console in Figure 3. Click on 'Start all services' and then check the 'W' icon on your toolbar. If the 'W' icon is green it means that all services are running. If it is red it means that no services are running. If it is orange it means that some services are running. If everything was installed correctly you should see a window such as the one in Figure 4.

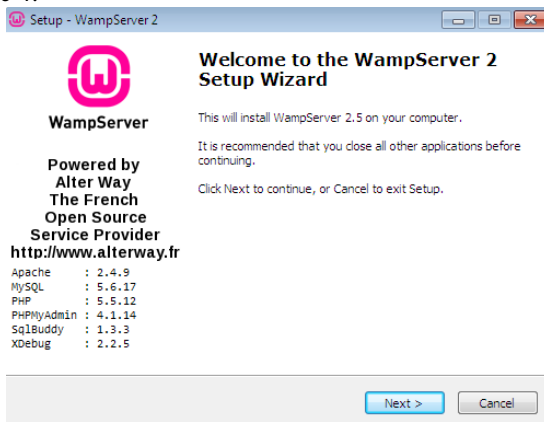


Figure 1: Welcome window to WAMP setup

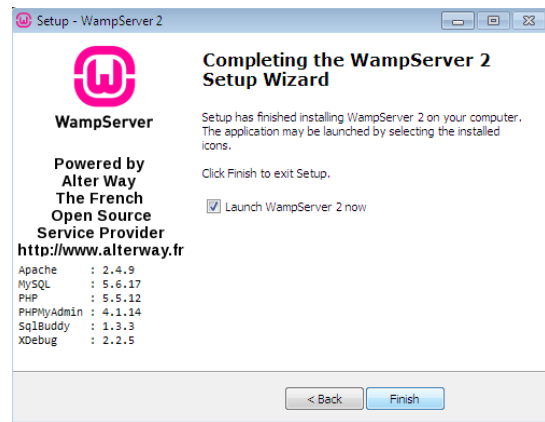


Figure 2: Finished installation

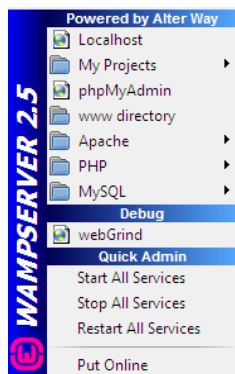


Figure 3: WampServer Management Console

# Web Servers

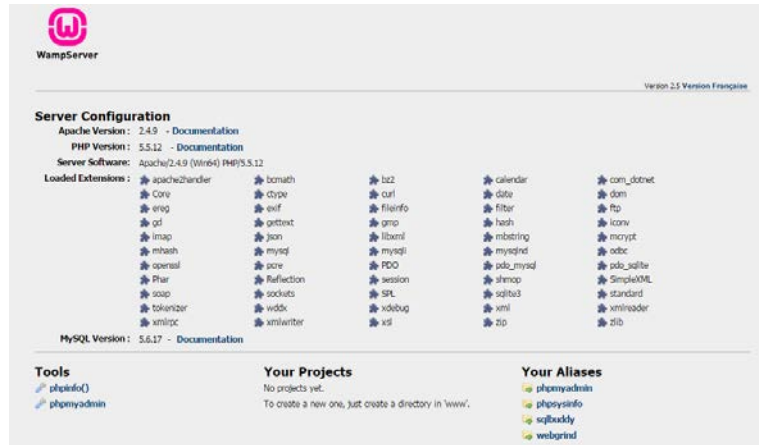


Figure 4: Localhost home screen

## 2.1.3 Setting up Server Passwords

In this step, you will learn how to create passwords for your server and also passwords to protect the files that you may want to share from your Web Server.

### i. Setting up MySQL and PHP Admin Password

On your local host home screen (Figure 4) click on 'phpmyadmin' under 'Tools'. The interface window opens showing the WAMP configuration page. At the bottom of this page, a message indicates that MySQL is running without a password (Figure 5).

Click on users and then check the box next to 'root localhost' and then click on 'Edit Privileges' (Figure 6). Scroll down to change the password and then click on 'Go' to save the changes. If you try to click on any other menu on the interface, for example on the SQL menu, you will get an error. Let us fix this by also changing the PHP admin password to match the MySQL password.

Open Windows Explorer. Navigate to the C:\wamp\apps\phpmyadminx.x.x\ folder (Figure 7). Inside that folder open **config.inc.php** – ideally using Notepad or any other html editor.

Search for the line `$cfg['blowfish_secret'] = ''`; – if you're using notepad it might be easier to just search for the word 'blowfish'. Change the line `$cfg['blowfish_secret'] = 'abcdef'`; to `$cfg['blowfish_secret'] = 'my passphrase'`; where **my passphrase** is your own password – **not the same one that you specified for root in MySQL**.

Now search for the phrase `['auth_type'] = 'config'`. Change 'config' to 'cookie'. Now search for `$cfg['Servers'][$i]['password'] = ''`; Replace the " with 'mysql-password'; where mysql-password is the MySQL password you created earlier.

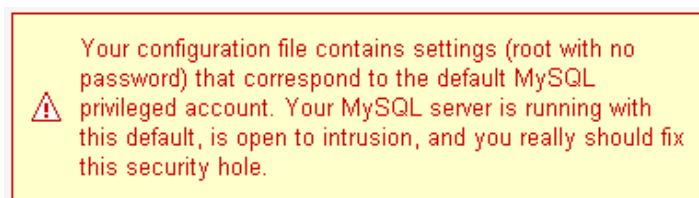
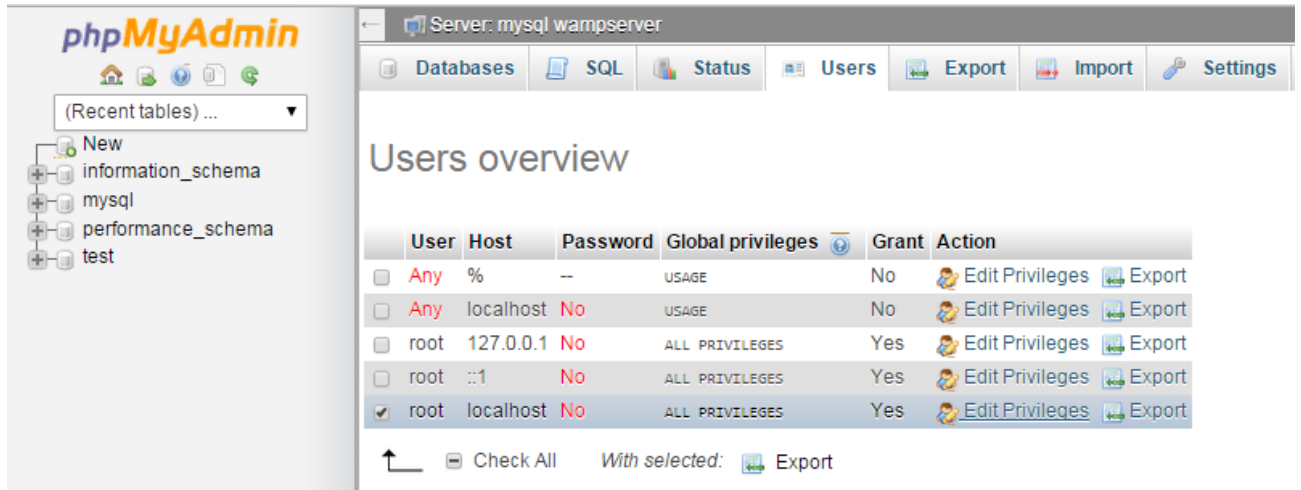
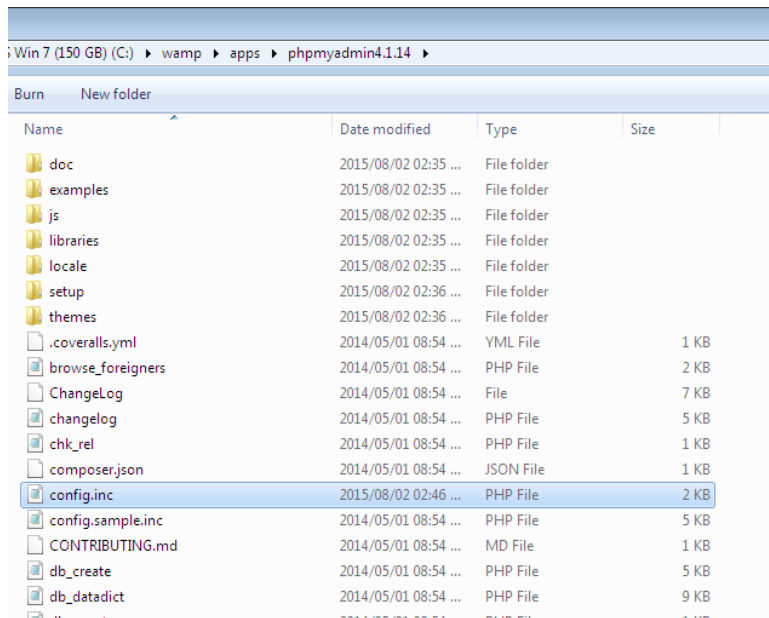


Figure 5: No Password set for WampServer

## Web Servers



**Figure 6: Changing root user privilege**



**Figure 7: Access config.inc folder**

Save the changes you have made and exit out of the editor. Go back to your toolbar where wampserver is located and click on 'restart all services'. Refresh localhost on your browser. Click on 'phpmyadmin'. You should now be prompted for a username and password. Your username is 'root' (since we did not change it from the default one) and your password is the MySQL password that you created.

### ii. Setting up a password to access files stored in the Web Server

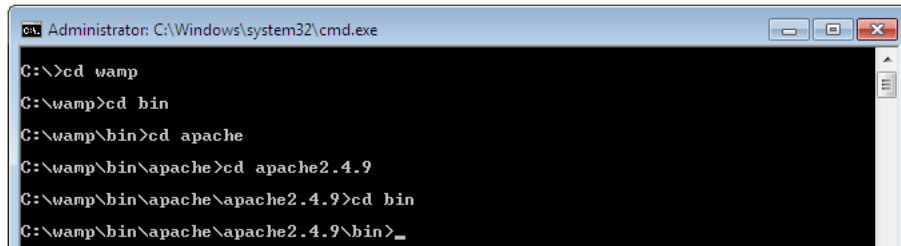
Now in case you want to share files from your server, you do not want just anyone to be able to access the files. So let us password-protect your files. For example, assume that you would like to store music files on your server and share them with others. First, decide where you would like to store your music files. In this illustration, the folder 'www' found in C:\wamp is used to store the music directory. We are assuming that you want your users to be able to access any folders that are stored inside the 'www' folder.

Next, using a command prompt, access the bin directory in the Apache folder (Figure 8). Then type:

## Web Servers

```
htpasswd -c "C:\wamp\my-password-file.txt" username
```

**my-password-file.txt** is the file where you create the password to access the 'www' folder. **username** is the username that you create to access the 'www' folder. Pick a username of your choice. **Note that the password file is not created inside the music folder.** After you press enter, you will be prompted to enter and re-enter a password. Create a password of your choice. This is the password that will be associated with the username you have just created, and the combination will be used to access the 'www' folder.



```
Administrator: C:\Windows\system32\cmd.exe
C:\>cd wamp
C:\wamp>cd bin
C:\wamp\bin>cd apache
C:\wamp\bin\apache>cd apache2.4.9
C:\wamp\bin\apache\apache2.4.9>cd bin
C:\wamp\bin\apache\apache2.4.9\bin>_
```

**Figure 8: Accessing Apache folder via command prompt**

Now we want to apply the login to your music folder. Open a new file in a plain text editor like Notepad. Copy and paste the following into it:

```
AuthType Basic
```

```
AuthName "This is a private area, please log in"
```

```
AuthUserFile "c:\wamp\my_password_file.txt"
```

```
AuthGroupFile /dev/null
```

```
require valid-user
```

You can replace the message **'this is a private area, please log in'** with your own security message that you would like users to see. Save this file in the 'www' folder which is our server root folder. There are two important things to note when saving this file:

1. Save this file as `.htaccess` (including the dot).
2. If using an html editor such as Notepad put quotation marks around the name, thus `"htaccess"`. This way, it will not be saved as a text file.

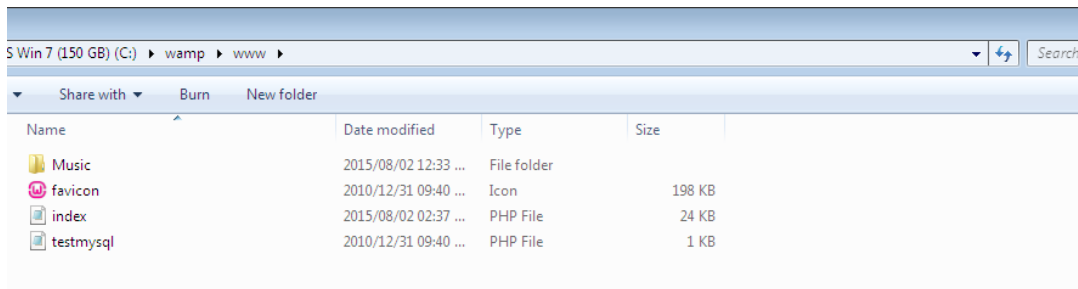
Now when you refresh your localhost on the browser, you should be prompted for login details.

### 2.1.4 Testing Applications

#### Sharing files stored on your WampServer

If you would like to share the music files that you have stored on your Web Server, simply create a directory called 'music' inside the 'www' folder (Figure 9). Put the music files that you would like to share in the 'music' folder. Type <http://localhost/music/> on your address bar and you should be able to see a listing of your music.

## Web Servers



**Figure 9: Music folder to be shared**

Take note of your computer's IP address. Go to your wampserver toolbar, left-click and click 'Put online'. Your server can now be accessed on the Web.

Share the address <http://xxx.xxx.xx.xxx/music/> with someone else to try on another computer. 'xxx' represents the numbers in your IP address. The user will be prompted for the login details that you setup in the last section. Note that this may not work if you are sharing the address with someone outside of your network who is behind a firewall.

We shall soon see how to write HTML files that can be accessed via the Apache HTML server.

The next section illustrates how to set up an Apache Tomcat server on Windows.

### Testing a 'Hello World' Application

Let us try a simple example to test that the websites we will create will work on this web server. Open Notepad and type the following code to print 'hello world' and save it as `hello.php` in the 'www' folder. Then access this file from your browser using your ip address or `localhost/hello.html`. You should get a 'hello world' output on your browser.

```
<html>
<head>
  <title>HTML Test</title>
</head>
<body>
  Hello world
</body>
</html>
```

## 2.2 Apache Tomcat on Windows

*Apache Tomcat* is a Java-capable HTTP server, which is able to execute special Java programs known as Java Servlet and Java Server Pages (JSP). It is also able to execute HTML files just like Apache HTTP.

### 2.2.1 Requirements

To illustrate the steps below a Windows 7 64-bit computer was used. The Windows computer had Internet access. Tomcat 8 was used for this installation. You need to have the latest Java JDK version installed. The recommended JDK version for Tomcat 8 is `jdk1.8`. Make sure that your computer is updated with this version. Go to [java.com](http://java.com) to download the latest version of Java.

The steps below are for a fresh installation of Apache Tomcat (assumes that Tomcat had not been installed before).

## 2.2.2 Tomcat Setup

### i. Download and Install Tomcat

Go to <http://tomcat.apache.org/> and download the latest version of Tomcat (Tomcat 9 at the time of writing this). Under Download on the left click on 'Tomcat 9' and under Binary Distribution click on 'Core' and you will see various packages. Click on the package applicable to you. For a Windows 64-bit computer click on '64-bit Windows zip'. Download this file.

Unzip the downloaded file to a directory of your choice. It is recommended not to unzip to the desktop as it is a difficult directory to locate from a command prompt. For this illustration a folder named 'tomcat' was created under drive D, hence D:/tomcat. The zipped file was extracted to this location. It is recommended to rename the unzipped file from the default name, for example, rename to 'apachetomcat' (Figure 10).

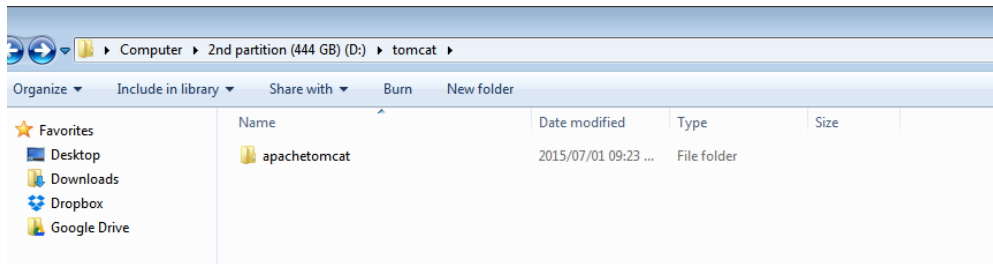


Figure 10: Creating folders to store tomcat files

### ii. Create an Environment Variable

First, take note of the directory into which JDK was installed. The default is "C:\Program Files\Java\jdk1.8.0\_{xx}", where {xx} is the latest upgrade number. It is important to verify your JDK installed directory before you proceed further. Start the command prompt and type 'set JAVA\_HOME' to check if the environmental variable had been set. If not, you will get the message 'Environment Variable JAVA\_HOME not set'. If JAVA\_HOME is set, check if it is set to your JDK installed directory correctly (For example in Figure 11). If not, proceed to set the environmental variable as below.

To set the environment variable JAVA\_HOME in Windows 7: Press "Start" button > Control Panel > System & Security > System > Advanced system settings > Switch to "Advanced" tab > Environment Variables > System Variables > "New" (or "Edit" for modification) > In "Variable Name", enter "JAVA\_HOME" > In "Variable Value", enter your JDK installed directory (e.g., "c:\Program Files\Java\jdk1.8.0\_51"). Click OK. To verify, **restart** the command prompt and type 'set JAVA\_HOME'. You should now see the output as in Figure 11.

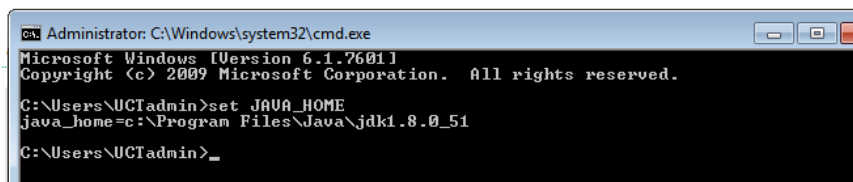


Figure 11: Checking environmental variables for JAVA\_HOME

### iii. Configure Tomcat Server

The Tomcat configuration files are located in the "conf" sub-directory of your Tomcat installed directory, e.g. "D:\tomcat\apachetomcat\conf". There are 4 configuration XML files: server.xml, web.xml, context.xml and tomcat-users.xml. Make a BACKUP of the configuration files before you proceed.

### Set the TCP Port Number

Use an HTML text editor (e.g., NotePad++) to open the configuration file "server.xml", under the "conf" sub-directory of Tomcat installed directory.

The default TCP port number configured in Tomcat is 8080, you may choose any number between 1024 and 65535, which is not used by an existing application. We shall choose 8888 in this article. Locate the following lines that define the HTTP connector, and change port="8080" to port="8888". Save the file and exit.

```
<Connector port="8888" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" />
```

### Enabling Directory Listing

Again, use an HTML text editor to open the configuration file "web.xml", under the "conf" sub-directory of Tomcat installed directory.

We shall enable directory listing by changing "listings" from "false" to "true" for the "default" servlet. Locate the following lines that define the "default" servlet; and change the "listings" from "false" to "true". Save and exit.

```
<servlet>
    <servlet-name>default</servlet-name>
    <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
    <init-param>
        <param-name>debug</param-name>
        <param-value>0</param-value>
    </init-param>
    <init-param>
        <param-name>listings</param-name>
        <param-value>true</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
```

### Enabling Automatic Reload

We shall add the attribute reloadable="true" to the <Context> element to enable automatic reload after code changes. Again, this is handy for test system but not for production, due to the overhead of detecting changes. Open context.xml and locate the <Context> start element, and change it to <Context reloadable="true">. Save and exit.

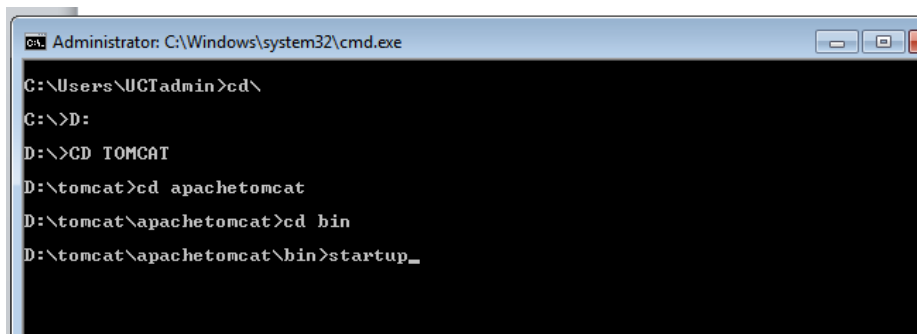
```
<Context reloadable = "true">
..... </Context>
```

#### iv. Start Tomcat Server

Launch a CMD shell. Set the current directory to the tomcat directory\bin", and run "startup.bat" as in Figure 12:



## Web Servers



```
Administrator: C:\Windows\system32\cmd.exe
C:\Users\UCTadmin>cd\
C:\>D:
D:\>CD TOMCAT
D:\tomcat>cd apachetomcat
D:\tomcat\apachetomcat>cd bin
D:\tomcat\apachetomcat\bin>startup_
```

Figure 12: Starting tomcat from command prompt

A new Tomcat console window appears (look out for the Tomcat's port number (double check that Tomcat is running on port 8888). Future error messages will be sent to this console. Output messages from related Java programs are also sent to this console. If you want to shut down the server type 'shutdown' in place of 'startup'.

Start a browser. Issue URL "http://localhost:8888" to access the Tomcat server's welcome page. For users on the other machines over the net, they have to use the server's IP address or DNS domain name or hostname in the format of "http://serverHostnameOrIPAddress:8888". Note that this may not work if you are sharing the address with someone outside of your network who is behind a firewall. If everything is setup correctly, you should see the screen in Figure 13.

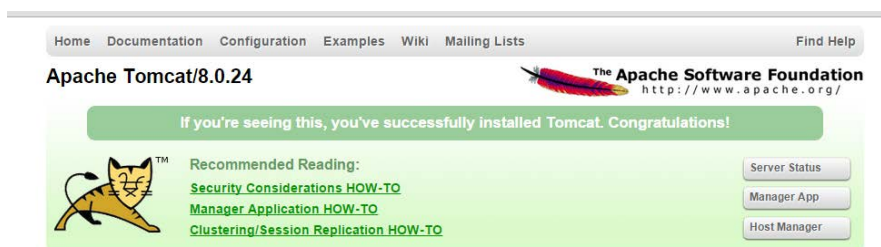


Figure 13: Tomcat home screen on local host

### Install Tomcat's Sample Web Application

Go to: <http://localhost:8888/docs/>

Click the link: "3. First web application"

Click "Example App" under contents on the left side of the screen.

Click on the link "here" to download their "Sample Application". (Download link: <http://localhost:8888/docs/appdev/sample/sample.war> )

Save to this location: C:\Tomcat\apachetomcat\webapps

Give the Tomcat container a minute and it will automatically extract the WAR file and create a Web Application called "Sample".

Test your install: <http://localhost:8888/sample>

If you wish to create your own directory under webapps, choose a *name* for your webapp. Let's call it "myapps".

Go to Tomcat's "webapps" sub-directory. Create the following directory structure for you webapp "myapps":

1. Under Tomcat's "webapps", create your webapp *root* directory "myapps" (i.e., "tomcat\apachetomcat\webapps\myapps").
2. Under "myapps", create a sub-directory "WEB-INF" (case sensitive, a "dash" not an underscore) (i.e., "tomcat\apachetomcat\webapps\myapps\WEB-INF").

## Web Servers

3. Under "WEB-INF", create a sub-sub-directory "classes" (case sensitive, plural) (i.e., "tomcat\apachetomcat\webapps\myapps\WEB-INF\classes").

Restart your tomcat server to pick up the changes. Then on your browser type <http://localhost:8888/myapps/>

You should see the directory listing of the directory "tomcat\apachetomcat\webapps\myapps", which shall be empty (provided you have enabled directory listing in web.xml earlier).

We shall soon see how to write HTML and PHP files that can be accessed via the Tomcat server.

### 2.2.3 Testing Applications

#### Sharing files stored on your Tomcat Server

If you would like to share the music files that you have stored on your Web Server, simply create a directory called 'music' insider the 'myapps' folder. Put the music files that you would like to share in the 'music' folder. Type <http://localhost:8888/myapps/music/> on your address bar and you should be able to see a listing of your music.

#### Testing a 'Hello World' Application on Tomcat Server

Let us try a simple example to test that the websites we will create will work on this web server. Open Notepad and type the following code to print 'hello world' and save is as hello.html in the 'myapps' folder. Then access this file from your browser using your ip address or <http://localhost:8888/myapps/hello.html>. You should get 'hello world' output on your browser.

```
<html>
<head>
<title>HTML Test</title>
</head>
<body>
Hello world
</body>
</html>
```

## 2.3 Lampserver and Apache HTTP on Ubuntu 15.04

In this section, you will learn how to set up a Web Server on Ubuntu 15.04. The steps in this section will illustrate how to use Apache HTTP. The next section will illustrate the setup for Apache Tomcat. LAMP stack is a group of open source software used to get web servers up and running. The acronym stands for Linux, Apache, MySQL or MariaDB, and PHP. Since I assume that your computer is already running Ubuntu, the Linux part is taken care of.

### 2.3.1 Requirements

To illustrate the steps below a 64-bit computer running Ubuntu 15.04 was used. The computer was connected to a local area network (LAN) with Internet access. You also need to know your server IP address. You can find out your IP address by right clicking the network icon in the notification area and clicking Connection Information, or by running *ifconfig -a* on the terminal. To log into your server, you will need to know the password for the "root" user's account. First, run the command below to update your server, before which you will be prompted to enter your root password.

```
sudo apt-get update
```

### 2.3.2 Installing Apache, PHP and MySQL

Run the commands below to install Apache2 Web Server and wait for completion of installation.

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

Run the commands below to start the Apache2 Web Server.

```
sudo service apache2 start
```

At this time, if you browse to the server using its IP address, you'll see Apache2 default page for Ubuntu as in Figure 14. This is how you also tell the server is up and functioning.

The next step is to install PHP as well as its module to enable PHP apps or web services to function. There are hundreds of PHP modules, but these few will get most web services started. To install PHP and other modules, run the commands below.

```
sudo apt-get install php5 php5-mysql php5-curl php5-gd php5-snmp php5-mcrypt  
php5-tidy php5-xmllrpc libapache2-mod-php5
```

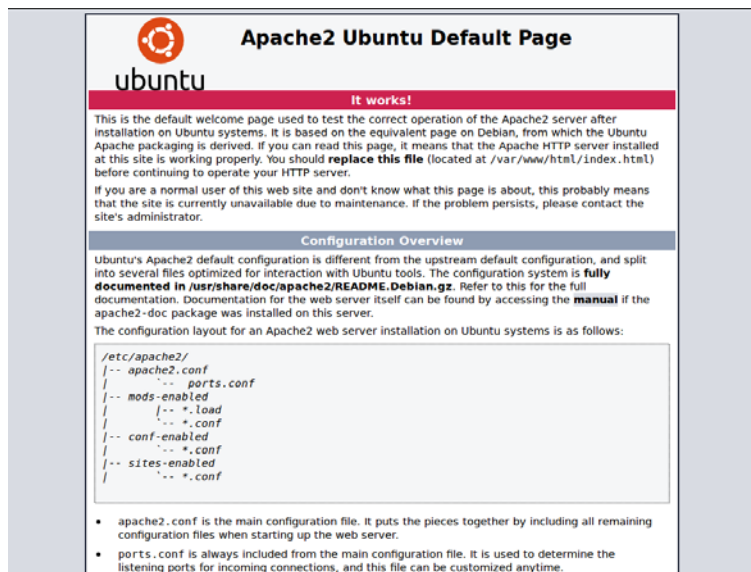


Figure 14: Successful Apache Installation

After installing the above modules, go to Apache root directory. It can be found at `/var/www/html` in Ubuntu. There create a file called `phpinfo.php`. Then in that file, add the lines below.


```
<?php  
phpinfo();  
>
```

Save the file, restart Apache and browser to the server IP address followed by `phpinfo.php`. (ex. <http://Your IP address/phpinfo.php>). There you'll find PHP information page such as Figure 15. This is how you also know PHP is functioning.

## Web Servers

Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/05-opcache.ini, /etc/php5/apache2/conf.d/10-pdo.ini, /etc/php5/apache2/conf.d/20-json.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226.NTS
PHP Extension Build	API20131226.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, tls, tlsv1.0, tlsv1.1, tlsv1.2
Registered Stream Filters	zlib.*, bzip2.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk

This program makes use of the Zend Scripting Language Engine:  
Zend Engine v2.6.0, Copyright (c) 1998-2014 Zend Technologies  
with Zend OPcache v7.0.4-dev, Copyright (c) 1999-2014, by Zend Technologies



**Figure 15: Successful PHP Installation**

The next step is to install MySQL. Run the commands below. Wait for completion.

```
sudo apt-get install mysql-server mysql-client
```

After installing the database server, you can start it using the commands below.

```
sudo systemctl start mysql
```

When it's started, run the commands below to configure the database server.

```
sudo mysql_secure_installation
```

When prompted, follow the options below and type your root password.

Next, choose Yes for the rest of the prompts until you're done.

- Enter current password for root (enter for none): Type root password
- Change the root password? **N**
- Remove anonymous users? **Y**
- Disallow root login remotely? **Y**
- Remove test database and access to it? **Y**
- Reload privilege tables now? **Y**

Restart Apache2 and you're done.

At this time, Apache2, PHP5 and other modules and MySQL database server should be installed and functioning. Your Ubuntu server is ready for any open source application that supports the LAMP stack.

You may wish to install PhpMyAdmin, which is a web interface through which you can easily manage/administer your MySQL/MariaDB databases. The installation can be completed with the following command:

```
sudo apt-get install phpmyadmin
```

Upon installation you will be asked to select the web server you are using. Select "Apache" and continue. Next you will be asked if you wish to configure phpmyadmin with dbconfig-common. Select "Yes". You need to perform one more step so that you can be able to access phpmyadmin from your Apache server. Run the following command.

```
sudo ln -s /usr/share/phpmyadmin /var/www/html
```

## Web Servers

If you now access phpmyadmin on your browser through <http://Your IP address/phpmyadmin> you should see the same window in Figure 16 and you can be able to log in using your MySQL username and password that you created.



Figure 16: Phpadmin home page

### 2.3.3 Testing Applications

#### Sharing files stored on your Tomcat Server

If you would like to share the music files that you have stored on your Web Server, create a directory called 'music' insider the 'var/www/html' folder.

Put the music files that you would like to share in the 'music' folder. Type <http://localhost/music/> on your address bar and you should be able to see a listing of your music.

#### Testing a 'Hello World' Application on Tomcat Server

Let us try a simple example to test that the websites we will create will work on this web server. Type the following command on the terminal to open the *gedit* editor.

```
sudo gedit
```

In the *gedit* editor, type the following code to print 'hello world' and save is at hello.html under var/www/html Then access this file from your browser using your ip address or localhost/hello.html. You should get 'hello world' as output in your browser.

```
<html>
<head>
<title>HTML Test</title>
</head>
<body>
Hello world
</body>
</html>
```

## 2.4 Apache Tomcat on Ubuntu 15.04

Make sure that Java JDK is installed on your machine. For this example, JDK-7 was installed. First, head-on-over to the Apache Tomcat 8 Download site. Then, under the heading 8.0.28 (the current version as of November 2015), or whichever is the newest version at the time you read this chapter, you'll see Binary Distributions. Under Binary Distributions you'll see Core and then under Core, you will see tar.gz. Right click on tar.gz and copy the URL link location.

## Web Servers

In the terminal use the URL you copied thus:

```
wget http://apache.is.co.za/tomcat/tomcat-8/v8.0.28/bin/apache-tomcat-8.0.28.tar.gz
```

After the download completes, decompress the file using the following command:

```
tar xvzf apache-tomcat-8.0.28.tar.gz
```

Now, move the file into a proper location using the following command. I am moving it to a folder called *opt*.

```
mv apache-tomcat-8.0.28 /opt
```

Now let's set the environment variables in *.bashrc* by typing the following command:

```
vim ~/.bashrc
```

Once in the vim editor, type 'i' in order to enter the insert mode. Add this information to the end of the file. After that press **esc** on the keyboard to go back to normal mode. Next press ':' (the colon) and this will drop the cursor to the bottom of the terminal. Here you save your changed by typing 'w' and press enter, and then you can type 'q' to quit vim.

```
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
export CATALINA_HOME=/opt/apache-tomcat-8.0.28
```

Make the changes effective by running the following command:

```
./~/.bashrc
```

Tomcat should now be installed and configured for your server. To activate Tomcat, run the following script on the terminal. Note that the commands are case sensitive.

```
$(CATALINA_HOME)/bin/startup.sh
```

You can verify that Tomcat is installed correctly by typing <http://localhost:8080> in your browser. You should see the window indicating successful installation.

### 2.4.1 Testing Applications

#### Sharing files stored on your Tomcat Server

If you would like to share the music files that you have stored on your Web Server, simply create a directory called 'music' insider the 'webapps' folder. Put the music files that you would like to share in the 'music' folder. Type <http://localhost/music/> on your address bar and you should be able to see a listing of your music.

#### Testing a 'Hello World' Application on Tomcat Server

Let us try a simple example to test that the websites we will create will work on this web server. Type the following command on the terminal to open the *gedit* editor.

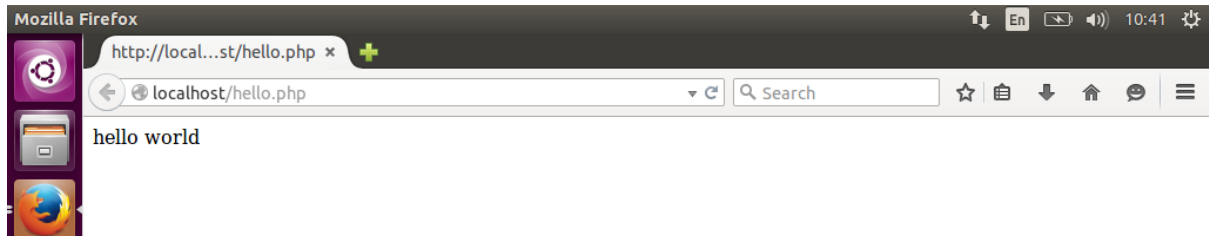
```
sudo gedit
```

In the *gedit* editor, type the following code to print 'hello world' and save is at *hello.html* under the *webapps* folder. Then access this file from your browser using your ip address or *localhost/hello.html*. You should get the output in Figure 17.

```
<html>
<head>
<title>HTML Test</title>
</head>
```

## Web Servers

```
<body>  
Hello world  
</body>  
</html>
```



**Figure 17: Hello World Output**