Web Servers:

A Web Server is a server program running on a computer whose purpose is to serve Web Pages to other computer when required. Every computer on the Internet that contains a Web site will have a Web Server program.

The most common use of web servers is to **host** websites, but there are other uses such as gaming, data storage, running enterprise applications, handling email, FTP, or other web uses.

Many generic web servers also support server-side scripting using Active Server Pages (ASP), PHP, or other scripting languages. This means that the behaviour of the web server can be scripted in separate files, while the actual server **software** remains unchanged. Usually, this function is used to create HTML documents dynamically ("on-the-fly") as opposed to returning static documents.

Examples of Web Servers:

1. Apache Web Server
2. Microsoft Internet Information Server (IIS)
3. XAMPP (Bundle server)
4. WAMP (Bundle server)

Apache HTTP Server

The Apache HTTP Server commonly referred to as Apache, is a web server program not able for playing a key role in the initial growth of the World Wide Web (WWW). It became the first web server software to exceed the 100 million web site mile stone. Typically Apache is run on a Unix-like Operating system, and was developed for use on Linux.

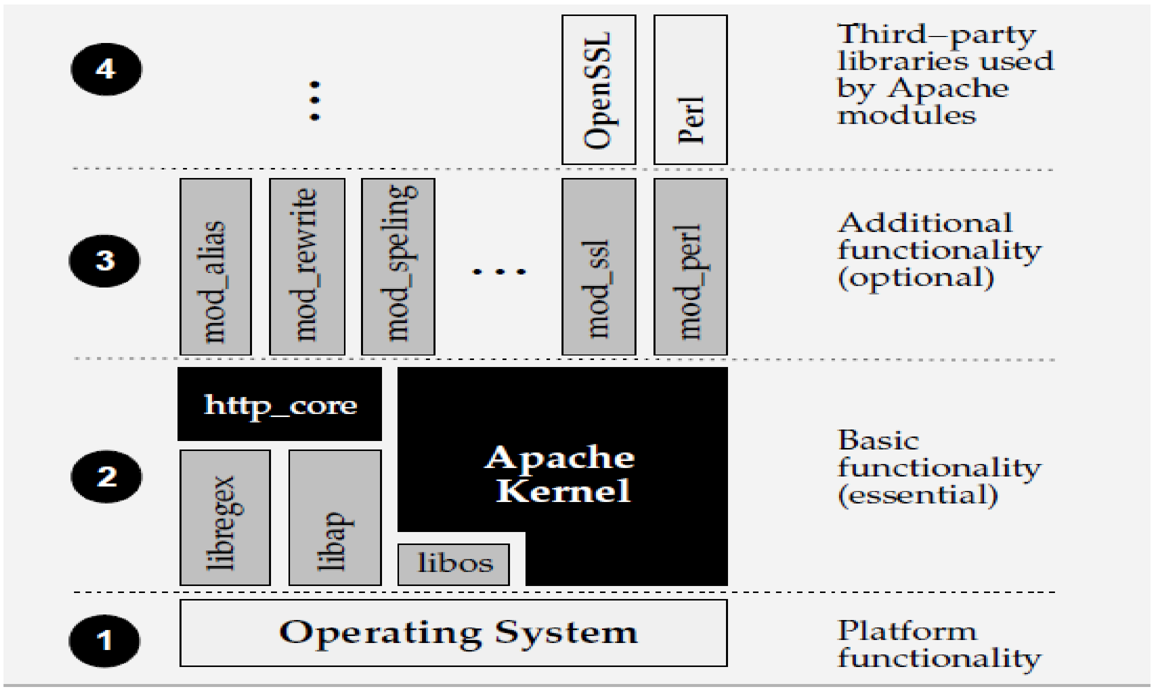
Apache is developed and maintained by an Open community of developers under the support and approval of the Apache Software Foundation (ASF). The application is available for wide variety of operating system, including UNIX, Free BDS, Solaris, Linux, Novel Netware, OSX, Microsoft Windows, OS/2 etc., Released under the Apache license, Apache is open-source software.

The main design goal of Apache is not to be the fastest Web server, Apache does have performance similar to other \high-performance" Web Servers. Instead of implementing a single architecture Apache provide a variety of Multi Processing Modules (MPMs) which allow Apache to run process-based, where compromises in performance need to be made, and the design of Apache is to reduce latency and increase throughput, relative to simply handling more requests, thusensuring consistent and reliable processing or requests within reasonable time frames.

Features of Apache:

- It implemented as compiled modules which extend the core functionality, thus the range from server-side programming support to authentication scheme.
- Password-protected pages for a multitude of users (It supports password authentication and digital certificate authentication).
- Customized error pages.
- Display of code in numerous levels of HTML, and the capability to determine at what level the browser can accept the content.
- Virtual hosting allows one Apache installation to serve many different actual Websites.
- Usage and error logs in multiple and customizable formats
- Directory Index directives to multiple files.
- URL aliasing or rewriting with no fixed limit

- Robustness and security



Architecture of the Apache Web Server

Microsoft Internet Information Server (IIS)

It is the second most popular Web Server software. It consists of Services including File Transfer Protocol (FTP), Hyper Text Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP) and others that enable a Windows machine to manage Websites. The latest version (IIS 7.6) also includes various modules for security, logging compression and diagnostics.

Because of IIS is provided for Windows systems only, the choice to use IIS necessitates the choice of a Windows Server and therefore increases running costs. Windows is also prone to more malware attacks, and has a reputation as a less secure server option.

But this presents a problem if you'd like to develop and even deploy your PHP-driven website on a Windows server running Microsoft's IIS web server. In recent years, Microsoft, in collaboration with Zend Technologies Ltd., has made great strides towards boosting both the stability and performance of PHP running on both Windows and IIS.

Features:

- IIS has a modular architecture. Modules, also called extensions, can be added or removed individually so that only modules required for specific functionality have to be installed.
- Security Module: Used to perform many tasks related to security in the requesting-processing pipeline (Authentication Scheme, URL authentication)
- Content Module: Used to perform tasks related to content in the requesting-processing pipeline (Such as processing requests for static pages, returning default page etc.,)
- Compression Module: Used to perform tasks related to compression in the requesting-processing pipeline (Such as compression responses, performing pre-compression of staticcontent.)
- Caching Module: Used to perform tasks related to caching in the requesting-processing pipeline (Such as storing processed information in the memory on the server and using cached content in subsequent request for the same resource.)
- Logging and Diagnostics Module: Used to perform tasks related to Logging and Diagnostics in the requesting-processing pipeline (Such as passing information and processing status to

HTTP.sys for logging, reporting events, and tracking requests currently executing in worker processes.)

- IIS 7.5 includes additional security features: Client-certificate mapping, IP security, Request filtering, URL authentication.

XAMPP (Bundle Server)

XAMPP is a free and open-source cross platform Web Server Solution stack package, consisting mainly of Apache HTTP Server, MySQL database, and interpreter for scripts written in the PHP and Perl programming languages.

X: Cross-Platform
A: Apache
M: MySQL
P: PHP
P: Perl

Officially, XAMPP's designers intended it for use only as a development tool, to allow Website designers and programmers to test their work on their own computer without any access to the Internet. To make this as easy as possible many important security features are disabled by default. XAMPP sometimes used to actually Server Web Pages on the World Wide Web.

Note: XAMPP is also provided support for creating and manipulating databases in MySQL and SQL Lite among others.

Benefits:

- Self contained, multiple instances of XAMPP can exist on a single computer, and any given instance can be copied from one computer to another.
- It automatically starts at system logon.
- You can start and stop Web Server and database stack with one command.
- Run in back ground.
- XAMPP is portable so you can carry it around on a thumb drive.
- The security settings are strict by default, nobody but you will be able to access the WebServer.
- PHP error reporting is enabled by default, which helps when debugging scripts.

WAMP (Bundle Server)

WAMP is the bundle of Apache, MySQL and PHP for Windows. These are the things you needto run a dynamic web sites on your computer in Windows. i.e equal to XAMPP.

WAMPs are packages of independently-created programs installed on computers that use aMicrosoft Windows operating system.

Benefits:

- Scripting language that can manipulate the information held in database and generate Web pages dynamically each time the content is requested by browser.
- Other packages are also included like phpMyAdmin which provides a GUI for database manager.

Some of the bundle servers are:

LAMP:Linux, **A**pache, **M**ysql, **P**HP.
SAMP:Solaris, **A**pache, **M**ysql, **P**HP.
MAMP:Mac OS, **A**pache, **M**ysql, **P**HP.

1.4 Installation of Web Servers

Installing Apache and PHP on Windows:

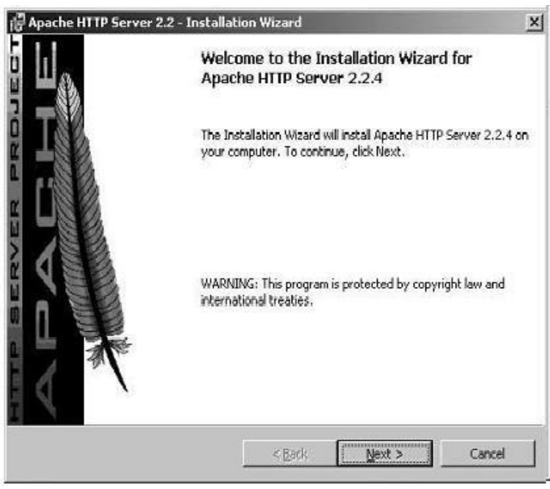
Apache needs to be installed and operational before PHP and MySQL

1. Download the Apache 2.x Win32 MSI installer binary. It's downloadable

fromhttp://httpd.apache.org/.

Select the –Download from a mirror‖ link on the left side of the page and download the best available version. A mirror is a download location. The file that you save to your desktop will be named similarly to apache2.2.4-win32-x86-nossl.msi (the exact version number will vary).

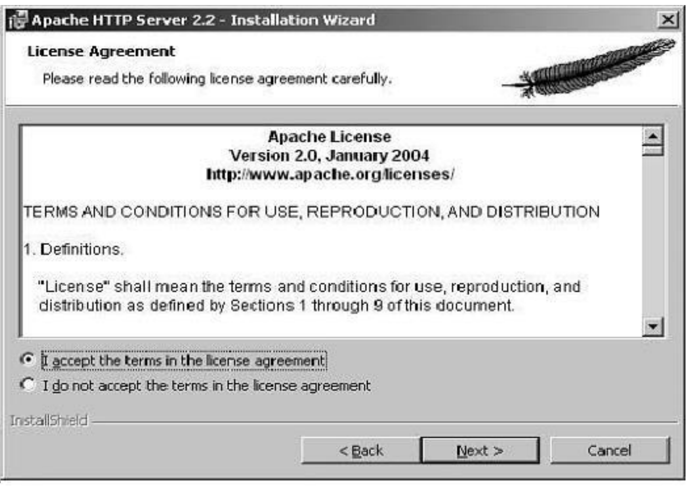
2. Install Apache using the Installation Wizard. Double-click the MSI installer file on your desktop, and you see the installer shown in Figure 1.3


Apache HTTP Server 2.2 - Installation Wizard
HTTP SERVER PROJECT APACHE
< Back
Next >
Cancel

Welcome to the Installation Wizard for Apache HTTP Server 2.2.4

The Installation Wizard will install Apache HTTP Server 2.2.4 on your computer. To continue, click Next.

WARNING: This program is protected by copyright law and international treaties.



License Agreement

Please read the following license agreement carefully.

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

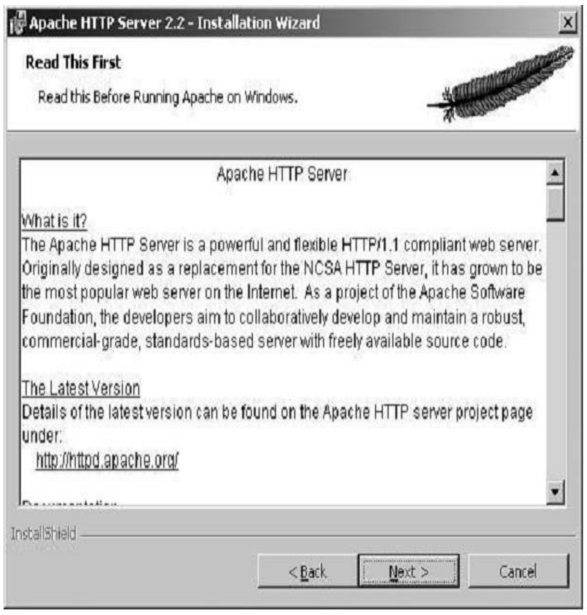
"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

[✓ I accept the terms in the license agreement]
I do not accept the terms in the license agreement

Figure 1.3 *Figure 1.4*

3. Accept the license terms by clicking the radio button shown in Figure 1.4. Click Next.

4. You'll see a Read This First box, as shown in Figure 1.5. Additionally, this window offersa number of excellent resources related to the web server. Click Next.



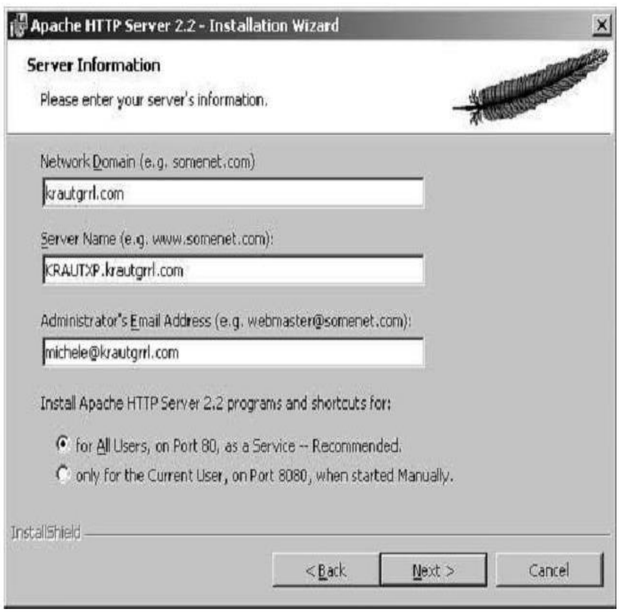
Read This First

Read this Before Running Apache on Windows.

Apache HTTP Server

What is it?
The Apache HTTP Server is a powerful and flexible HTTP/1.1 compliant web server. Originally designed as a replacement for the NCSA HTTP Server, it has grown to be the most popular web server on the Internet. As a project of the Apache Software Foundation, the developers aim to collaboratively develop and maintain a robust, commercial-grade, standards-based server with freely available source code.

The Latest Version
Details of the latest version can be found on the Apache HTTP server project page under:
http://httpd.apache.org/



Server Information

Please enter your server's information.

Network Domain (e.g. somenet.com)
krautgrrl.com

Server Name (e.g. www.somenet.com):
KRAUTXP.krautgrrl.com

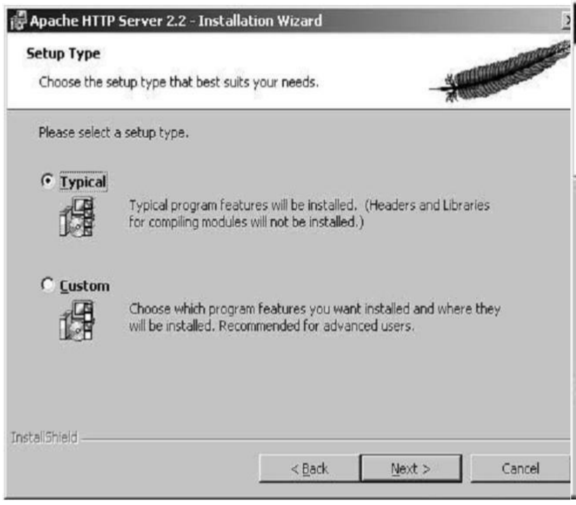
Administrator's Email Address (e.g. webmaster@somenet.com):
michele@krautgrrl.com

Install Apache HTTP Server 2.2 programs and shortcuts for:
[✓ for All Users, on Port 80, as a Service -- Recommended.]
only for the Current User, on Port 8080, when started Manually.

Figure 1.5 *Figure 1.6*

5. In the dialog shown in Figure 1.6, enter all pertinent network information. Click Next.

6. In the next screen, shown in Figure 1.7, select the setup type. The Typical install will work for your purposes. Click Next.


Apache HTTP Server 2.2 - Installation Wizard

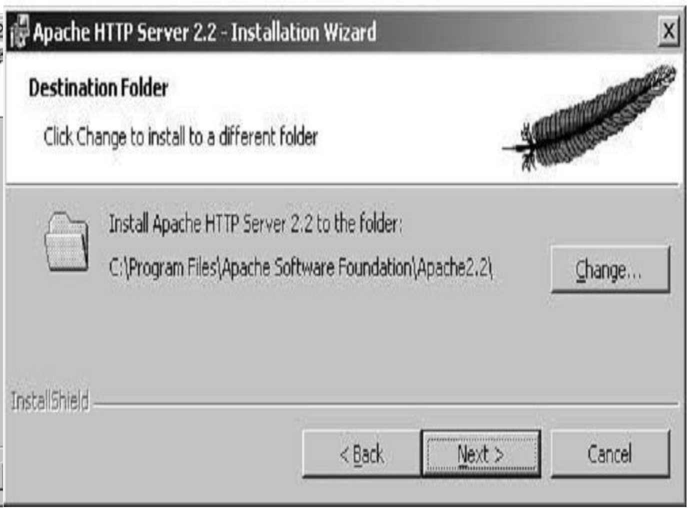
Setup Type

Choose the setup type that best suits your needs.

Please select a setup type.

(●) Typical — Typical program features will be installed. (Headers and Libraries for compiling modules will not be installed.)

( ) Custom — Choose which program features you want installed and where they will be installed. Recommended for advanced users.


Apache HTTP Server 2.2 - Installation Wizard

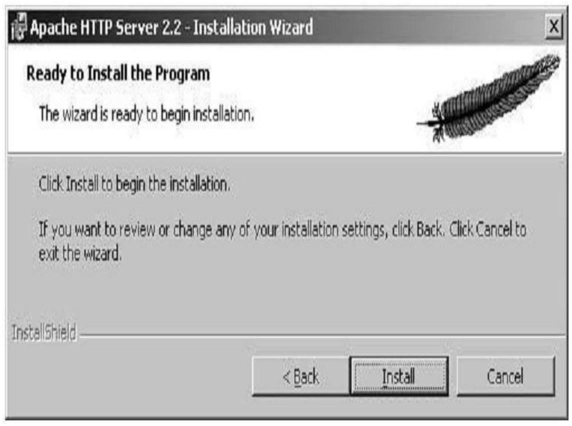
Destination Folder

Click Change to install to a different folder

Install Apache HTTP Server 2.2 to the folder:
C:\Program Files\Apache Software Foundation\Apache2.2\

Figure 1.7 *Figure 1.8*

7. Accept the default installation directory, as shown in Figure 1.8. Click Next.
8. As Figure 1.9 shows, it's time to begin the installation. Click Install. The installer installs a variety of modules, and you will see some DOS windows appear and disappear.

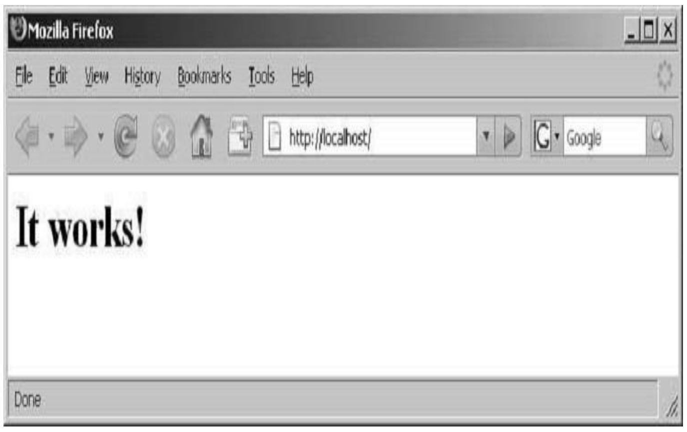

Apache HTTP Server 2.2 - Installation Wizard

Ready to Install the Program

The wizard is ready to begin installation.

Click Install to begin the installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.



It works!

Figure 1.9 *Figure 1.10*

9. Click Finish when the installer is done.
10. Test your installation by entering **http://localhost/** in your browser's location field. Remember, local host is just the name that translates to the IP address **127.0.0.1**, which is always the address of the local computer.
11. After entering the URL in your browser, the default Apache page displays, which is similar to the one shown in Figure 1.10. The installation was successful if you see the text –It works!".This page may be different depending on which version of Apache you install.

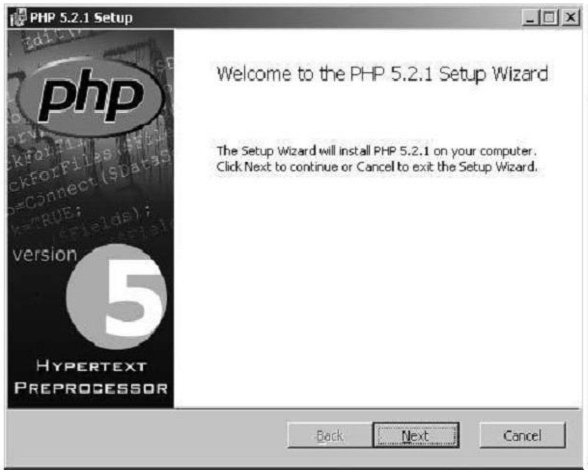
Generally, if you see text that doesn't mention an error, the installation was successful.

Installing PHP

Go to **http://www.php.net/downloads.php** to download the latest version of PHP; both binaries and source code can be found on this web site.

1. The file that you save to your desktop will be named similarly to **php-5.2.1-win32-installer.msi** (the exact version number will vary).

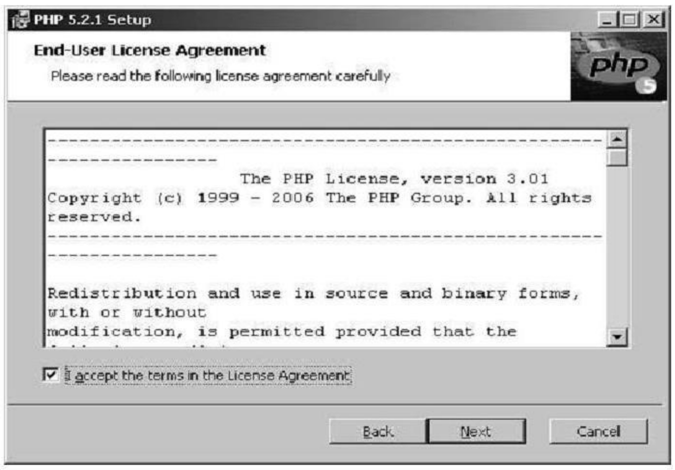
2. Install PHP using the Installation Wizard. Double-click the MSI installer file on your desktop, and you‘ll see the installer shown in Figure 1.11.



PHP 5.2.1 Setup

Welcome to the PHP 5.2.1 Setup Wizard

The Setup Wizard will install PHP 5.2.1 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.



PHP 5.2.1 Setup

End-User License Agreement

Please read the following license agreement carefully

```
--------------------------------------------------------------------
                The PHP License, version 3.01
Copyright (c) 1999 - 2006 The PHP Group. All rights
reserved.
--------------------------------------------------------------------

Redistribution and use in source and binary forms,
with or without
modification, is permitted provided that the
```

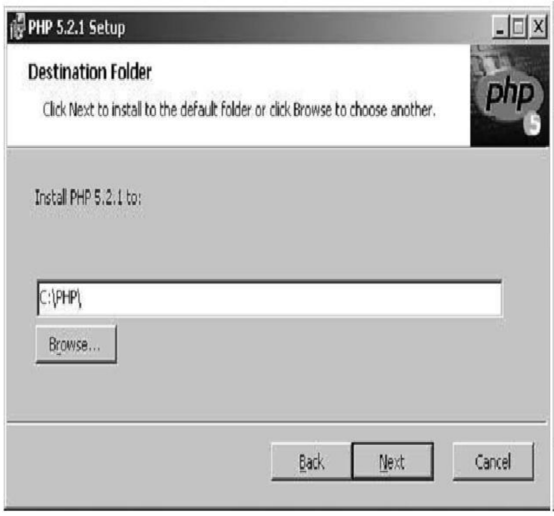
[✓] I accept the terms in the License Agreement

Figure 1.11 *Figure 1.12*

3. Click Next. The License Terms dialog appears as shown in Figure 1.12

4. Click the checkbox to accept the licensing terms. Click Next.

5. The Destination Folder dialog appears (see Figure 1.13). Select the destination folder. You may use the default of **C:\Program Files\PHP** or **C:\PHP** (used in this material that modifythe PHP configuration files assume C:\PHP). Click Next.



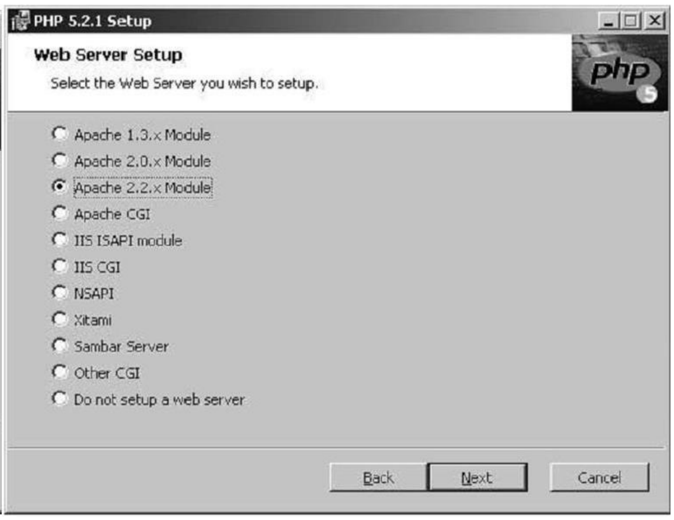
PHP 5.2.1 Setup

Destination Folder

Click Next to install to the default folder or click Browse to choose another.

Install PHP 5.2.1 to:

C:\PHP\



PHP 5.2.1 Setup

Web Server Setup

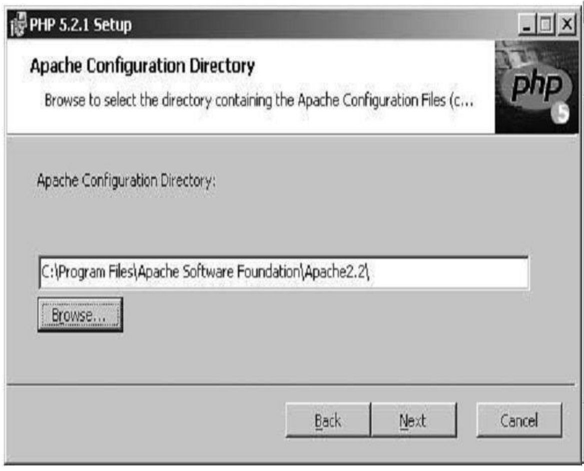
Select the Web Server you wish to setup.

- ○ Apache 1.3.x Module
- ○ Apache 2.0.x Module
- ● Apache 2.2.x Module
- ○ Apache CGI
- ○ IIS ISAPI module
- ○ IIS CGI
- ○ NSAPI
- ○ Xitami
- ○ Sambar Server
- ○ Other CGI
- ○ Do not setup a web server

Figure 1.13 *Figure 1.14*

6. The Web Server Setup dialog appears as shown in Figure 1.14. Select –Apache 2.2.xModule" and click Next. If you were using a different web server like IIS, you could select that option here.

7. The Apache Configuration Directory dialog specifies where you installed Apache so that the installer can set up the Apache configuration to use PHP for you. It should be similar to **C:\Program Files\Apache Software Foundation\Apache2.2\**, as shown in Figure 1.15.


PHP 5.2.1 Setup

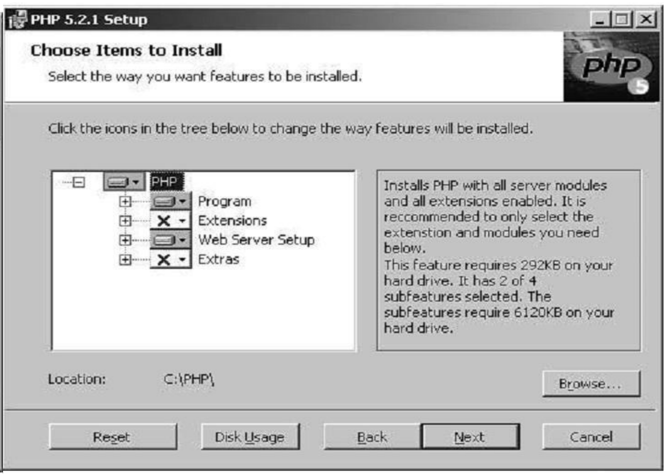
Apache Configuration Directory

Browse to select the directory containing the Apache Configuration Files (c...

Apache Configuration Directory:

C:\Program Files\Apache Software Foundation\Apache2.2\

Figure 1.15


PHP 5.2.1 Setup

Choose Items to Install

Select the way you want features to be installed.

Click the icons in the tree below to change the way features will be installed.

Installs PHP with all server modules and all extensions enabled. It is reccommended to only select the extenstion and modules you need below.
This feature requires 292KB on your hard drive. It has 2 of 4 subfeatures selected. The subfeatures require 6120KB on your hard drive.

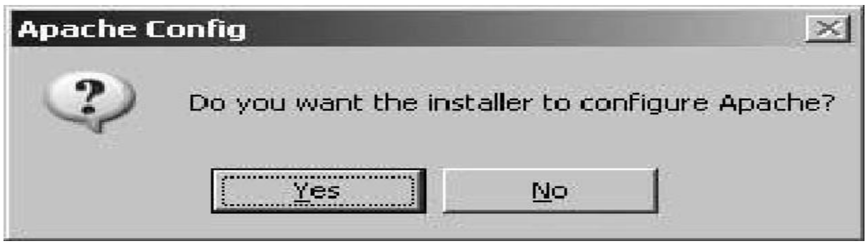
Location: C:\PHP\

Figure 1.16

8. Figure 1.16 shows the –Choose Items to Install" dialog. The defaults on this dialog areall OK. If you changed the base install directory, you may also need to change it here.Click Next.

9. Click Install on the –Ready to install" screen to confirm the installation.

10. Click Yes to confirm configuring Apache when the dialog shown in Figure 1-17 appears.


Apache Config

Do you want the installer to configure Apache?

Figure 1.17: Dialog confirming that the installer will configure Apache

11. Click OK on the Apache Config dialog to acknowledge the successful Apache update for –**httpd.conf**‖.

12. Click OK on the Apache Config dialog to acknowledge the successful Apache update for mime.types.

13. The Successful Installation dialog appears.

14. Restart the Apache server by selecting Start → All Programs → Apache HTTP Server 2.x.x → Control Apache Server → Restart, so that it can read the new configuration directives that the PHP installer placed in the **httpd.conf** configuration file. This file tells Apache to load the PHP process as a module. Alternatively, in the system tray, double-click the Apache icon and click the Restart button.

Installing Apache for Linux/UNIX

To download the Apache distribution for Linux, start at the Apache Server Web site, **http://httpd.apache.org/**,and follow the link to Download. The current version is 2.2.4, and I prefer *.tar.gz files, so the file used as an example throughout this section is httpd-2.2.4.tar.gz.

1. Type **cp httpd-2.2.4.tar.gz /usr/local/** and press Enter to copy the Apache installation file to the **/usr/local/src** directory.
2. Go to **/usr/local/src** by typing **cd/usr/local/src** and pressing Enter.
3. Unzip the Apache installation le by typing **gunzip httpd-2.2.4.tar.gz** and pressing Enter.
4. Extract the files by typing **tar -xvf httpd-2.2.4.tar** and pressing Enter. A directory structure will be created, and you‘ll be back at the prompt. The parent directory will be**/usr/local/src/httpd-2.0.49/**.

5. Enter the parent directory by typing **cd httpd-2.2.4** and pressing Enter.
6. Type the following and press Enter to prepare to build Apache:

```
./configure --prefix=/usr/local/apache2 --enable-module=so
```

The configuration script will run through its process of checking your configuration and creatingmake files, and then it will put you back at the prompt.

7. Type **make** and press Enter. This second step of the installation process will produce many lines of output on your screen. When it is finished, you will be back at the prompt.
8. Type **make** install and press Enter. This final step of the installation process will again produce many lines of output on your screen. When it is finished, you will be back at the prompt.

If your installation process produces any errors up to this point, go through the process again or check the Apache Web site for any system-specific notes. In the next section, you'll make some minor changes to the Apache conguration file before you start Apache for the first time.

Configuring Apache on Linux

To run a basic installation of Apache, the only changes you need to make are to the server name, which resides in the master configuration file called **httpd.conf**. This file lives in the**conf** directory, within the Apache installation directory. So if your installation directory is**/usr/local/apache2/**, the configuration files will be in **/usr/local/apache2/conf/**.

To modify the basic configuration, most importantly the server name, open the **httpd.conf**file with a text editor and look for a heading called Main server configuration. You will find two important sections of text.

We are going to change the values in the configuration file so that Apache knows where to find things and who to send complaints to. The **ServerAdmin**, which is you, is simply the e-mail address that people can send mail to in reference to your site. The **ServerName** is what Apache uses to route incoming requests properly.

1. Change the value of **ServerAdmin** to your e-mail address.
2. Change the value of **ServerName** to something accurate and remove the preceding # so that the entry looks like this:

```
ServerName somehost.somedomain.com
```

3. Save the file.

Installing PHP for Linux

To download the PHP source distribution, visit the Downloads page at the PHP Web site:**www.php.net/downloads.php**.

1. The current source code version is 6.0.0, and that version number will be used in thefollowing steps.
2. Once downloaded to your system, type **cp php-6.0-dev.tar.gz /usr/local/src/** and press Enter to copy the PHP source distribution to the **/usr/local/src/** directory.
3. Go to **/usr/local/src/** by typing **cd /usr/local/src/** and pressing Enter.
4. Unzip the source file by typing **gunzip php-6.0-dev.tar.gz** and pressing Enter.
5. Extract the files by typing **tar -xvf php-6.0-dev.tar** and pressing Enter. This will createa directory structure and then put you back at the prompt. The parent directory will be**/usr/local/src/php-6.0.0/**.
6. Enter the parent directory by typing **cd php-6.0-dev** and pressing Enter.
7. Type the following and press Enter to prepare to build PHP:

```
./configure --prefix=/usr/local/php5 --with-mysql=/usr/local/mysql/
--with-apxs2=/usr/local/apache2/bin/apxs
```

The configuration script will run through its process of checking your configuration andcreating makefiles and then will put you back at the prompt.

8. Type **make** and press Enter. This second step of the installation process will produce many lines of output on your screen. When it is finished, you will be back at the prompt.
9. Type **make** install and press Enter. This final step of the installation process will produce many lines of output on your screen. When it is finished, you will be back at the prompt.

Now, to get a basic version of PHP working with Apache, all you need to do is to make a few modifications to the **httpd.conf**file.

Configuring Apache to Use PHP

The installation process will have placed a module in the proper place within the Apache directory structure. Now you must make some modifications to the **httpd.conf**file before starting up Apache with PHP enabled.

1. Open the **httpd.conf**file in your text editor of choice.
2. Look for the following line, which will have been inserted into the file by the installation process:

```
LoadModule php6_module modules/libphp6.so
```

You want this line to be uncommented, so ensure that it is (as shown).

3. Look for the following lines:

```
# AddType allows you to add to or override the MIME configuration #
filemime.types for specific file types.
#AddType application/x-tar .tgz
```

4. Add to these lines the following:

```
AddType application/x-httpd-php .phtml .php
```

5. Save and close the **httpd.conf**file.

Installing IIS and PHP on Windows

<u>Installing IIS Services:</u>

a) Install windows in to your machine.
b) After installing windows, go to control panel.
c) In control panel, double click add/remove programs.
d) In the left part of the window, you can see the option add/remove windows component .click on add/remove window components.
e) Now, you can see the internet information services as a checkbox, initially unchecked.
f) Select that checkbox and click ok, automatically the system will ask for windows CD (windows XP only ie., Win7 won‘t ask for any CD),insert windows CD & click ok. It will install IIS on your system.
g) After installation, open control panel, double click on administration tools, double click on internet services manager icon. This opens the internet information services window (or)type "inetmgr" at start & run command.
h) Place the documents that will be requested from IIS either in the default directory i.e., c:\onetpub\wwwroot)in a virtual directory.
i) A virtual directory is an alias for an existing directory that resides on the local machine or on the network.
j) In the IIS window, the left pane contains the web server directory structure.
k) The name of the machine running IIS is listed under IIS.
l) Clicking the # symbol to the left of the machine name displays default FTP site, default web site, default SMTP virtual server.
m) FTP site is a file protocol site. the default website is an HTTP site.
n) The default SMTP virtual server allows you to create a simple mail transfer

protocol(SMTP)sever for sending electronic mail(e-mail).

o) Expand the default website directory by clicking '+'.The default web server sub directories are virtual directories.

p) In this directory, we will create a virtual directory for the HTTP web site. to create a virtual directory with in this directory, right click on default web site, select "new", then virtual directory. this starts the virtual directory creation wizard.

q) In the virtual directory alias dialog, enter the name for the virtual directory & click next.

r) In the website content directory dialog, enter the path for the directory containing the documents that client will view. click next.

s) The access permissions dialog presents the virtual directory security level choices, includes read, run script, execute ,write, browse.

<u>Installing & configuring PHP in windows with IIS Server</u>:-

1. Download php zip file from php.net
2. Install IIS in the system.
3. Extract all of the files from the downloaded zip file into c:\php.
4. Php5 includes a CGI Executable as well as he server module. the DLL's needed for their executable scan be found in the root of the php folder(c:\php).
5. phpts.dll needs to be available to the web browser to do this , you have 3 options.
   a) copy php5ts.dll to the web servers directly(c:/input/user).
   b) copy php5ts.dll to the windows system directory(sys32).
   c) add to path directory path environment variables.
6. Right click on my computer and choose properties.
7. Select advanced tab click the environment variables button.
8. In the system variables click on "path" select edit
9. Go to the end of the line add a semicolon(;) if there is not one already add c:\php and click ok.
10. Restart the computer once to effect the changes.
11. Now we want to setup config file for php, php.ini.
12. In c:\php you will find two files named php.ini_production, php.ini_development. selectphp.ini_production.
13. Rename it to php.ini.
14. Open php.ini search doc_root = c:\inetpub\wwwroot.
15. cgi.force_redirect = 0 and save it.

<u>Move to php with IIS</u>:-

1. Open IIS.
2. Expand the IIS directories, right click on default websites under home directory tab , set execute permissions to scripts only.
3. Click on configuration click and button.
4. Browse the path to php i.e..to insert php5 is api.dll. i.e..(c:\php\php5 is api.dll) set extension to php<u>ok.</u>
5. Under isapi filters, add, set filter name to php. Set executable to c:\php\php5 isapi.dll <u>ok</u>apply.
6. Open note pad type: <?php echo phpinto(); ?>. Save it as phpinfo.php in c:\inetpub\wwwroot.

Open web browser as type http://localhost/phpinfo.php.

Installing a XAMPP on Linux

If you know much about Linux, you may have already set up and installed PHP and MySQL. Ifnot, your best bet is probably to look at XAMPP for Linux, which is available athttp://apachefriends.org/en/xampp-linux.html.

The process is relatively simple. After downloading, go to a Linux shell and log in as thesystem administrator (root) by typing:**su**

Enter your system administration password. Many desktop Linux systems allow you to useyour personal account's password for the administration password. Some systems, includingthe popular Ubuntu, encourage you not to use su to log in as root, but to precede each systemadministration command with **sudo** instead. You'll know what to do if you've performed anyadministrative tasks on your system. Now extract the downloaded archive file to /opt with thefollowing command (inserting the appropriate filename if the version you downloaded is a laterversion):

```
tarxvfz xampp-linux-1.6.8a.tar.gz -C /opt
```

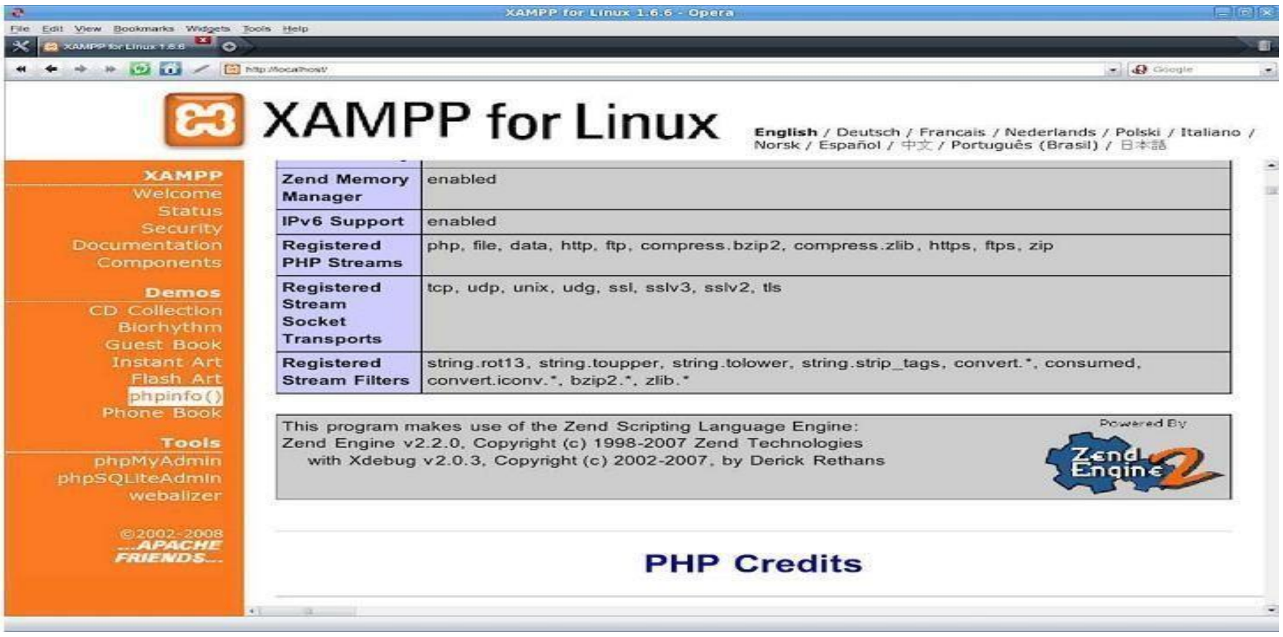
Any XAMPP version that was already installed will be overwritten by this command. Oncethe command finishes, XAMPP will be installed below the /opt/lampp directory. To start it,enter the following: **/opt/lampp/lampp start**

You should now see something like this on your screen:

```
Starting XAMPP 1.6.8a...
LAMPP: Starting Apache...
LAMPP: Starting MySQL...
LAMPP started.
```

Ready. Apache and MySQL are running.

Now you are ready to test the setup. Type the following URL into your web browsers addressbar:http://localhost


XAMPP for Linux 1.6.6 - Opera
File Edit View Bookmarks Widgets Tools Help
http://localhost/
XAMPP for Linux
English / Deutsch / Francais / Nederlands / Polski / Italiano / Norsk / Español / 中文 / Português (Brasil) / 日本語
XAMPP
Welcome
Status
Security
Documentation
Components
Demos
CD Collection
Biorhythm
Guest Book
Instant Art
Flash Art
phpinfo()
Phone Book
Tools
phpMyAdmin
phpSQLiteAdmin
webalizer
©2002-2008 ...APACHE FRIENDS...
Powered By Zend Engine 2

Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, data, http, ftp, compress.bzip2, compress.zlib, https, ftps, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, sslv2, tls
Registered Stream Filters	string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, convert.iconv.*, bzip2.*, zlib.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.2.0, Copyright (c) 1998-2007 Zend Technologies
with Xdebug v2.0.3, Copyright (c) 2002-2007, by Derick Rethans

PHP Credits

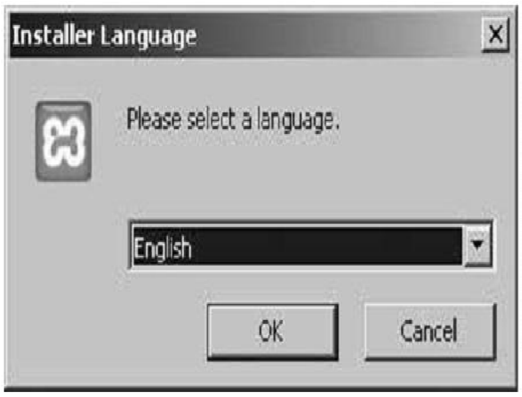
Figure 1.18: XAMPP for Linux, installed and running

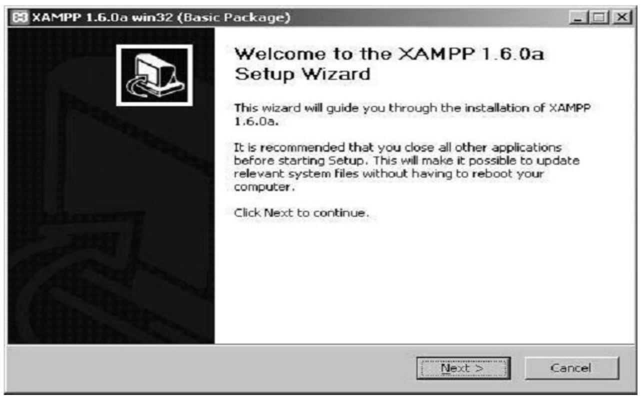
Installing XAMPP on Windows

The following steps cover installing XAMPP on Windows:

1. Download the Basic Package XAMPP MSI installer found at http://www.apachefriends.org/en/xampp-windows.html
2. Double-click the MSI installer file on your desktop, and you‘ll see the installer shown inFigure 1.19
3. Select English and click the OK button.

4. The Setup Wizard appears as shown in Figure 1.20. Click Next.
5. The dialog shown in Figure 1.21 is displayed. Click Next to accept the default installationdirectory.





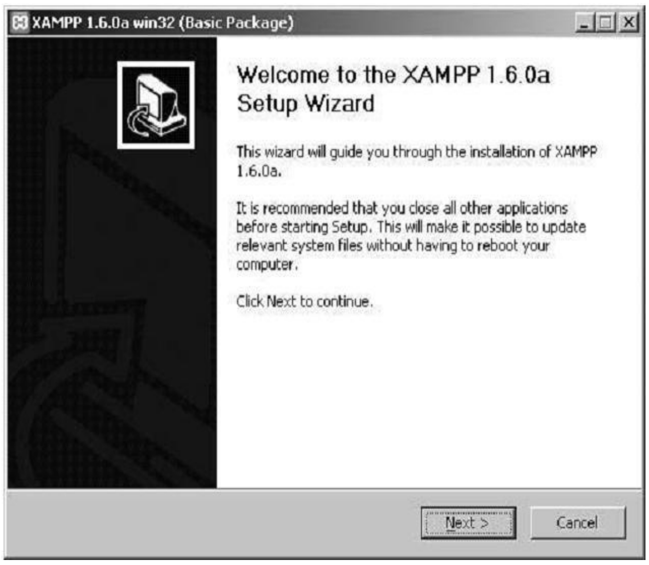
Welcome to the XAMPP 1.6.0a Setup Wizard

This wizard will guide you through the installation of XAMPP 1.6.0a.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 1.19 *Figure 1.20*



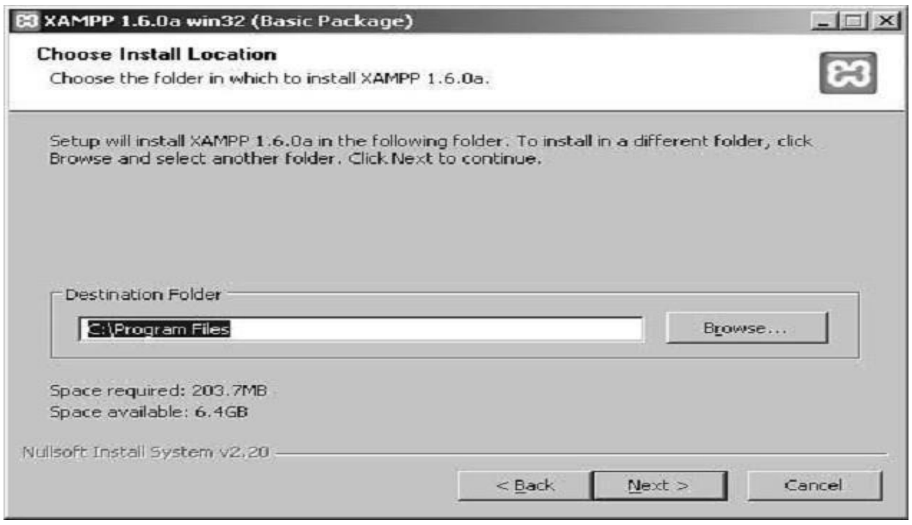
Welcome to the XAMPP 1.6.0a Setup Wizard

This wizard will guide you through the installation of XAMPP 1.6.0a.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 1.20: The Xampp Setup Wizard


XAMPP 1.6.0a win32 (Basic Package)
Browse...
Nullsoft Install System v2.20
< Back
Next >
Cancel

Choose Install Location
Choose the folder in which to install XAMPP 1.6.0a.

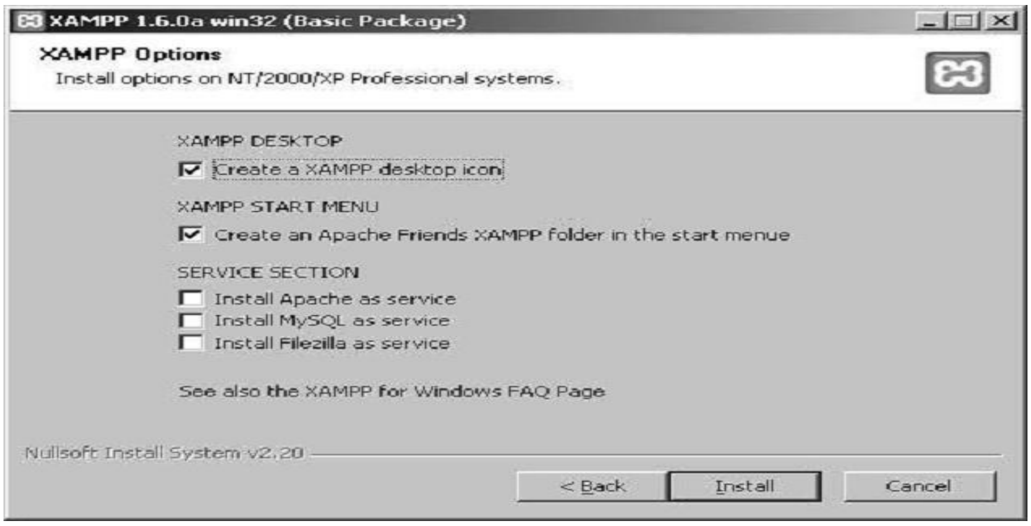
Setup will install XAMPP 1.6.0a in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder
C:\Program Files

Space required: 203.7MB
Space available: 6.4GB

Figure 1.21: Select the installation directory

6. The XAMPP Options dialog displays, as shown in Figure 1.22. Leave the Service Section checkboxes unchecked so you don‘t install the components as services; instead, you'll start them from the Control Panel. Click Install.



XAMPP 1.6.0a win32 (Basic Package)

XAMPP Options
Install options on NT/2000/XP Professional systems.

XAMPP DESKTOP
☑ Create a XAMPP desktop icon

XAMPP START MENU
☑ Create an Apache Friends XAMPP folder in the start menue

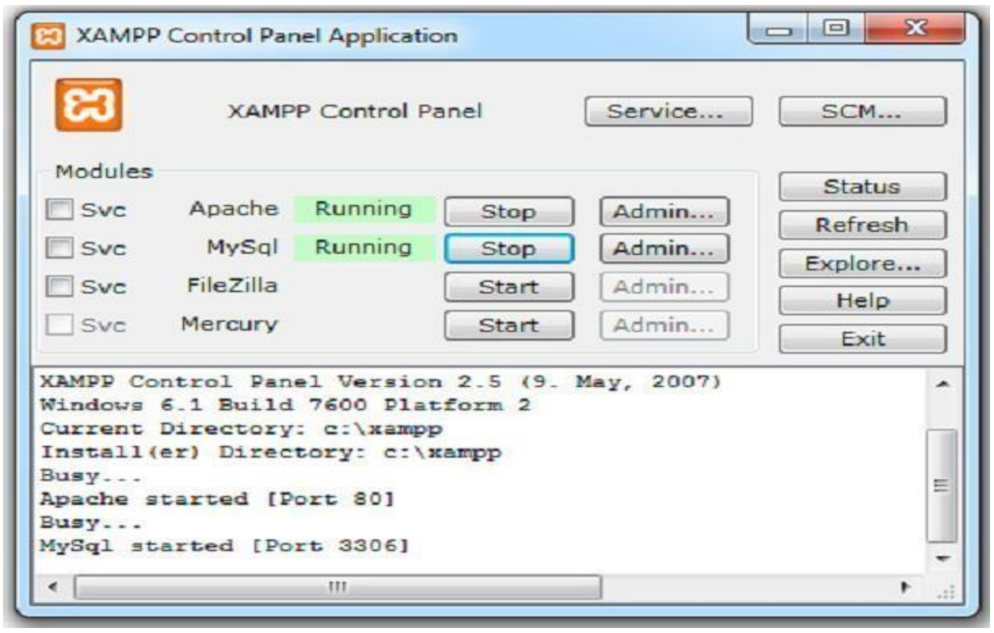
SERVICE SECTION
☐ Install Apache as service
☐ Install MySQL as service
☐ Install Filezilla as service

See also the XAMPP for Windows FAQ Page

Nullsoft Install System v2.20

Figure 1.22: Choose your installation options

7. The Completing the XAMPP Setup Wizard displays. Click Finish.
8. The option to start the Control Panel displays, Click Yes.
9. The Control Panel launches, as shown in Figure 1.23. The Control Panel can start and stop the services, as well as aid in their configuration.



XAMPP Control Panel Application

XAMPP Control Panel

Modules

☐ Svc	Apache	Running	Stop	Admin...
☐ Svc	MySql	Running	Stop	Admin...
☐ Svc	FileZilla		Start	Admin...
☐ Svc	Mercury		Start	Admin...

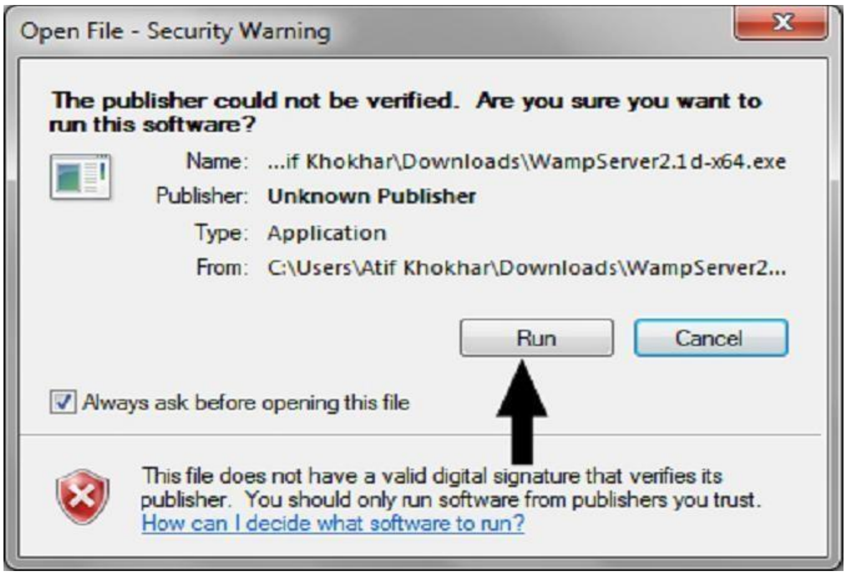
```
XAMPP Control Panel Version 2.5 (9. May, 2007)
Windows 6.1 Build 7600 Platform 2
Current Directory: c:\xampp
Install(er) Directory: c:\xampp
Busy...
Apache started [Port 80]
Busy...
MySql started [Port 3306]
```

Figure 1.23: The Control Panel starts and stops the components

Installing WAMP

If you are installing **WampServer 2.1d**, then these following steps will help you that how to install the WampServer 2.1d in your computer with windows 7. This server can be found fordownload at official web page WampServer.

1. It is the time to install WampServer on our windows. You will receive a Security Warningafter opening WampServer file. It is absolutely normal to run WampServer setup on windows.(Fig 1.24)


Open File - Security Warning

The publisher could not be verified. Are you sure you want to run this software?

Name: ...if Khokhar\Downloads\WampServer2.1d-x64.exe
Publisher: Unknown Publisher
Type: Application
From: C:\Users\Atif Khokhar\Downloads\WampServer2...

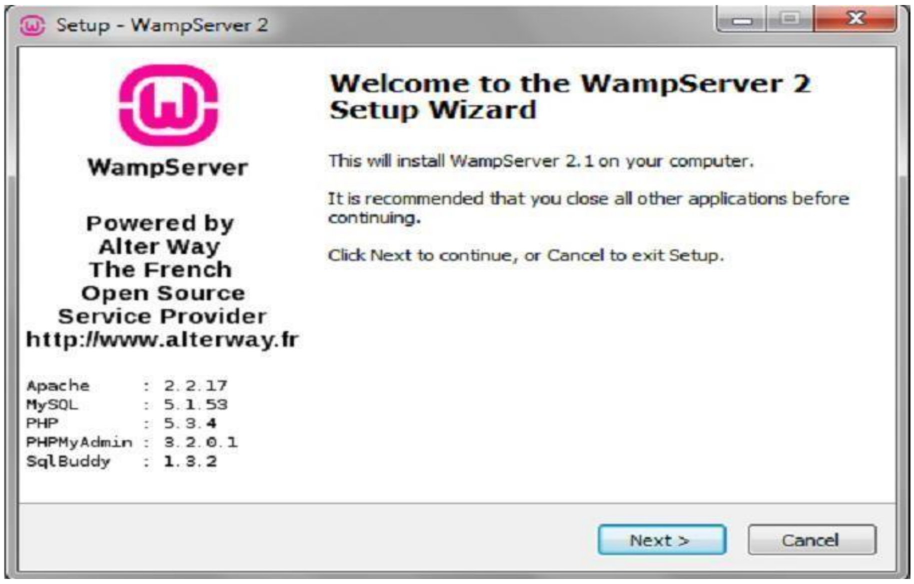
Run Cancel

☑ Always ask before opening this file

This file does not have a valid digital signature that verifies its publisher. You should only run software from publishers you trust.
How can I decide what software to run?

Figure 1.24: Instalation Starting of WampServer

2. You will see a standard setup wizard of windows after clicking Run button on securitywarningdialog(Fig 1.25)


Setup - WampServer 2

WampServer

Powered by
Alter Way
The French
Open Source
Service Provider
http://www.alterway.fr

```
Apache     : 2.2.17
MySQL      : 5.1.53
PHP        : 5.3.4
PHPMyAdmin : 3.2.0.1
SqlBuddy   : 1.3.2
```

Welcome to the WampServer 2 Setup Wizard

This will install WampServer 2.1 on your computer.

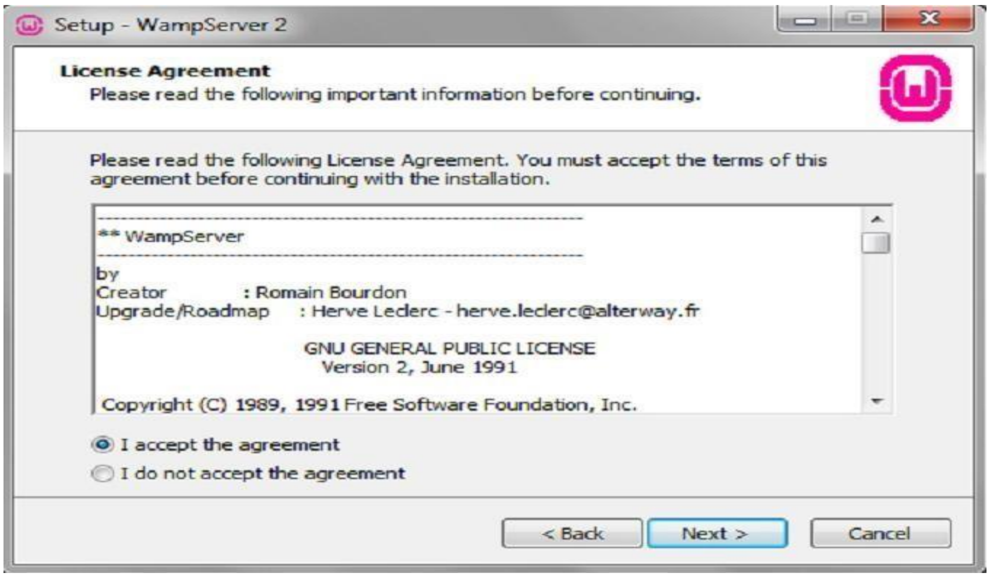
It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Next > Cancel

Figure 1.25: WampServer 2 Setup Wizard

3. You have to agree the license of WampServer before selecting installation destination atyour windows machine.(Fig. 1.26)


Setup - WampServer 2

License Agreement
Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

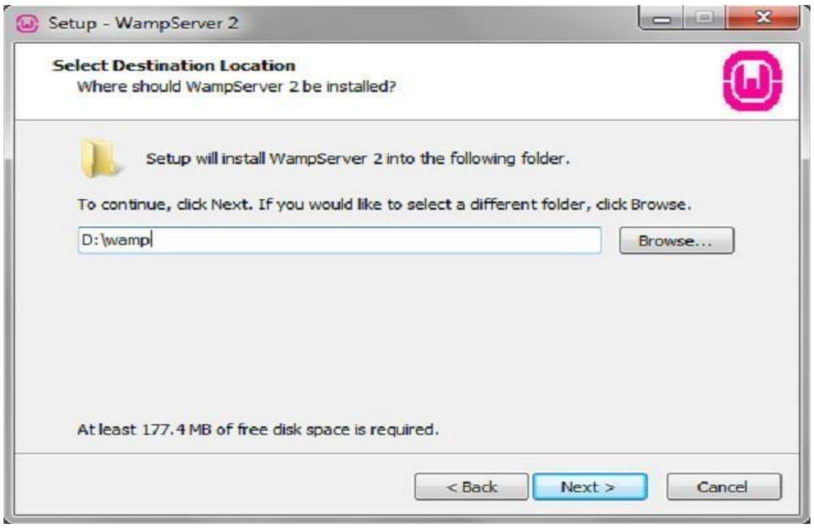
```
-----------------------------------------------
** WampServer
-----------------------------------------------
by
Creator           : Romain Bourdon
Upgrade/Roadmap     : Herve Leclerc - herve.leclerc@alterway.fr

                 GNU GENERAL PUBLIC LICENSE
                   Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.
```

[✓ I accept the agreement]
I do not accept the agreement

< Back Next > Cancel

4. It is very important step of WampServer installation. I will recommend installingWampServerat the drive other than Windows 7 installation. Suppose your Windows 7 is install in Cdrive so you should install WampServer on D, E or any other location in hard drive exceptC drive.

I am going to install WampServer in D drive. Now you can click on Next button afterselecting installation location for WampServer 2.1d.(Fig. 1.27)


Setup - WampServer 2

Select Destination Location
Where should WampServer 2 be installed?

Setup will install WampServer 2 into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

D:\wamp

At least 177.4 MB of free disk space is required.

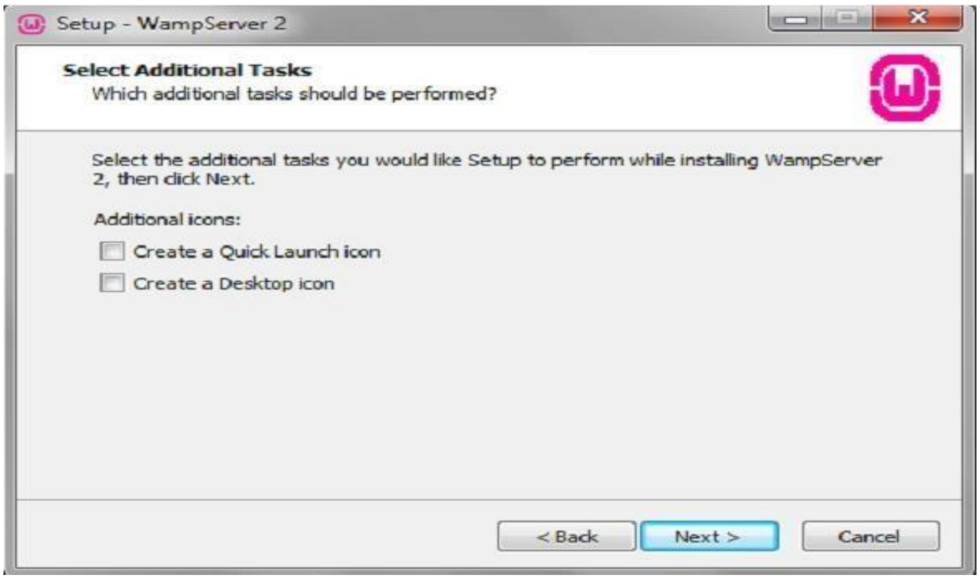
Figure 1.27: Select Destination Location of WampServer

5. When you click on the Next button then a Select Additional Tasks dialog will appear onyour screen, if you would like setup to perform while installing WampServer 2. You cancheck following options,

- Create a Quick Launch icon
- Create a Desktop icon

I have not interested to create any icon in the above locations, but you can do. You willbe at –Ready to Install" window after clicking Next button.(Figure 1.28)

6. Setup is now ready to begin installing WampServer 2.1d on your computer. Click on Install button to start installation of WampServer 2.1d.(Figure 1.29)


Setup - WampServer 2

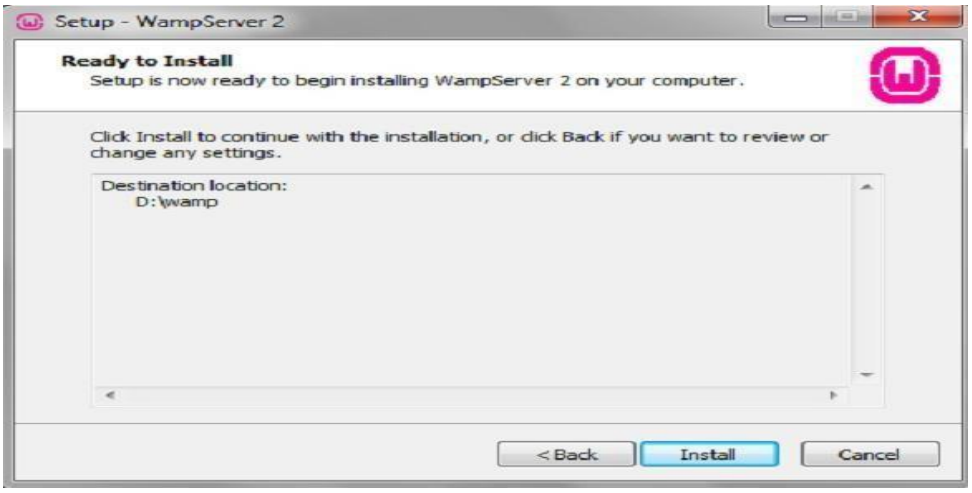
Select Additional Tasks
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing WampServer 2, then click Next.

Additional icons:

☐ Create a Quick Launch icon
☐ Create a Desktop icon

Figure 1.28: Select Additional Tasks



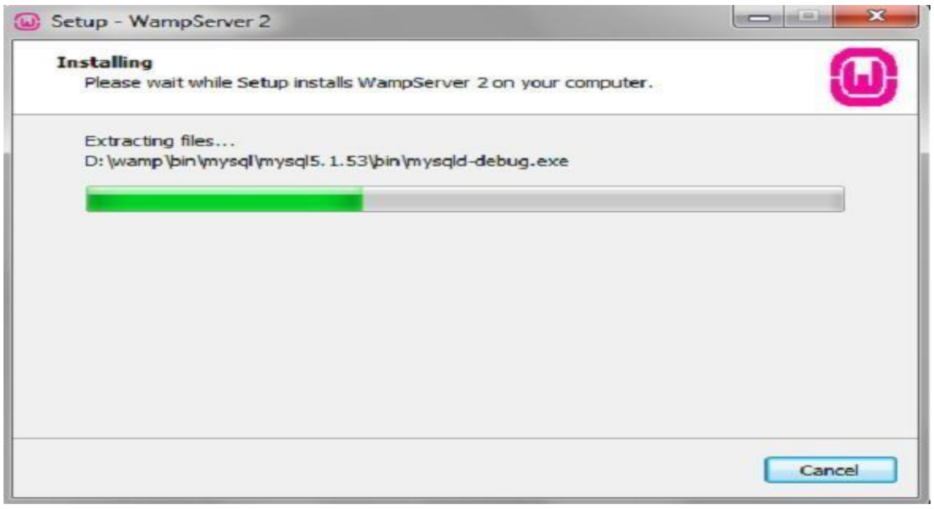
Ready to Install
Setup is now ready to begin installing WampServer 2 on your computer.

Click Install to continue with the installation, or click Back if you want to review or change any settings.

Destination location:
D:\wamp

Figure 1.29: WampServer 2.1d Ready to Install

7. Now your WampServer is starting to install in your computer.(Figure 1.30)

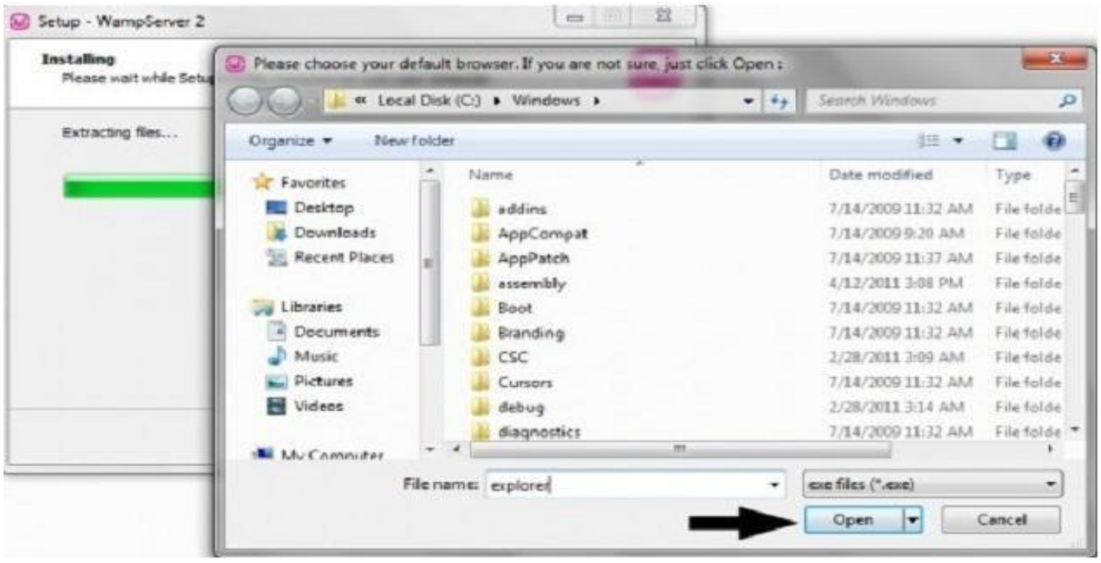


Installing
Please wait while Setup installs WampServer 2 on your computer.

Extracting files...
D:\wamp\bin\mysql\mysql5.1.53\bin\mysqld-debug.exe

Figure 1.30: WampServer Installing

8. You will receive a dialog for choosing your default browser for WampServer. You canchoose your favorite browser for WampServer as default, or simply click –Open" if you are not sure about the installation or executable files of your favorite browser.(Figure 1.31)



Please choose your default browser. If you are not sure, just click Open :

Name	Date modified	Type
addins	7/14/2009 11:32 AM	File folde
AppCompat	7/14/2009 9:20 AM	File folde
AppPatch	7/14/2009 11:37 AM	File folde
assembly	4/12/2011 3:08 PM	File folde
Boot	7/14/2009 11:32 AM	File folde
Branding	7/14/2009 11:32 AM	File folde
CSC	2/28/2011 3:09 AM	File folde
Cursors	7/14/2009 11:32 AM	File folde
debug	2/28/2011 3:14 AM	File folde
diagnostics	7/14/2009 11:32 AM	File folde

File name: explorer
exe files (*.exe)

Figure 1.31: Choice of Default Browser

9. WampServer installation has completed now and setup will guide you for Apache configurations in the next steps.(Figure 1.32)

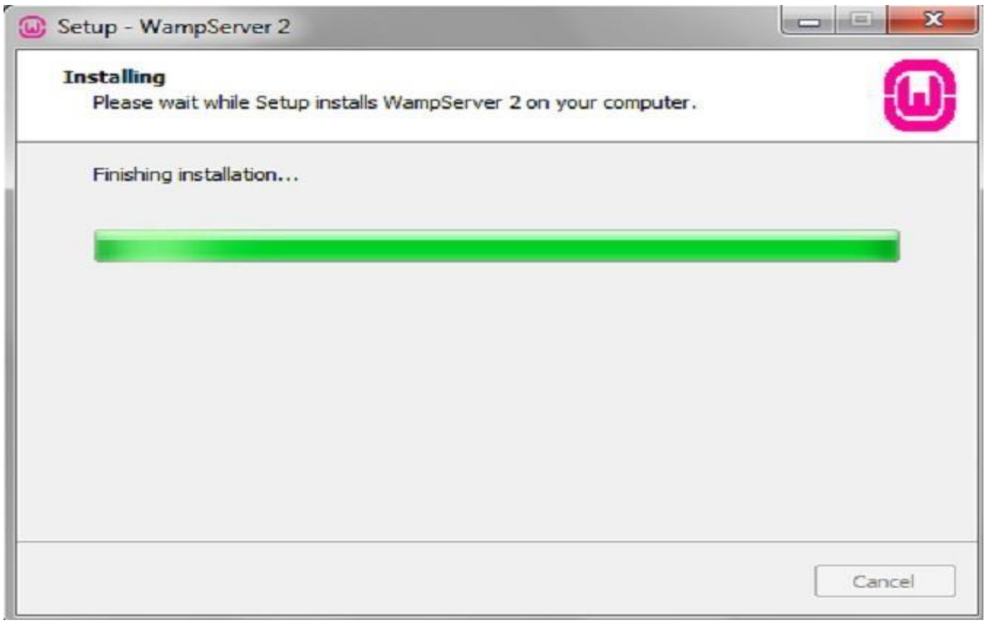

Setup - WampServer 2
Installing
Please wait while Setup installs WampServer 2 on your computer.
Finishing installation...
Cancel

Figure 1.32: Complete the Installation

10. You will notice a –Windows Firewall" standard dialog while configuring Apache byWampServer.(You may not observe this, if your windows firewall is not active). Click on –Allow Access"by leaving default options as such to proceed for PHP mail parameters.(Figure 1.33)

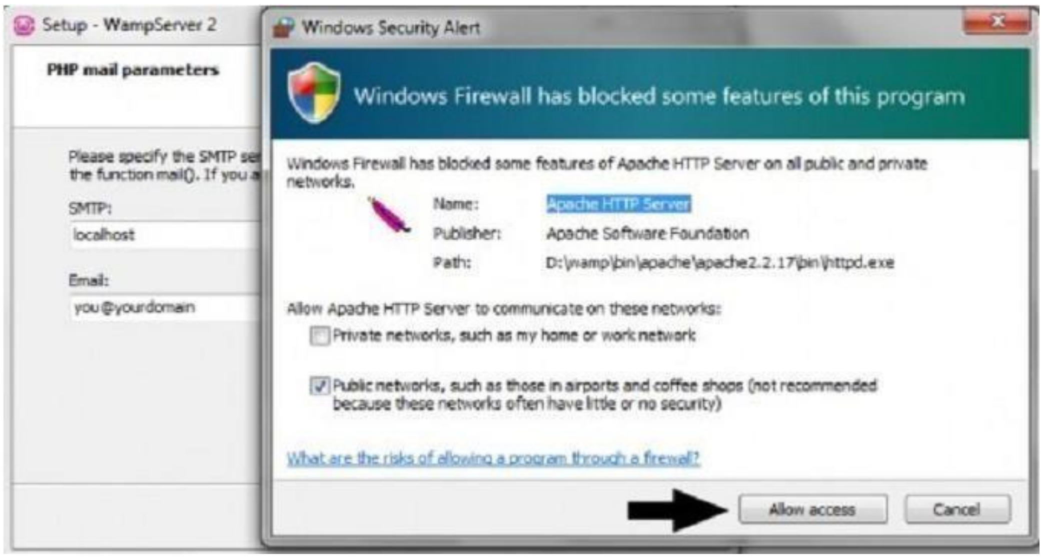
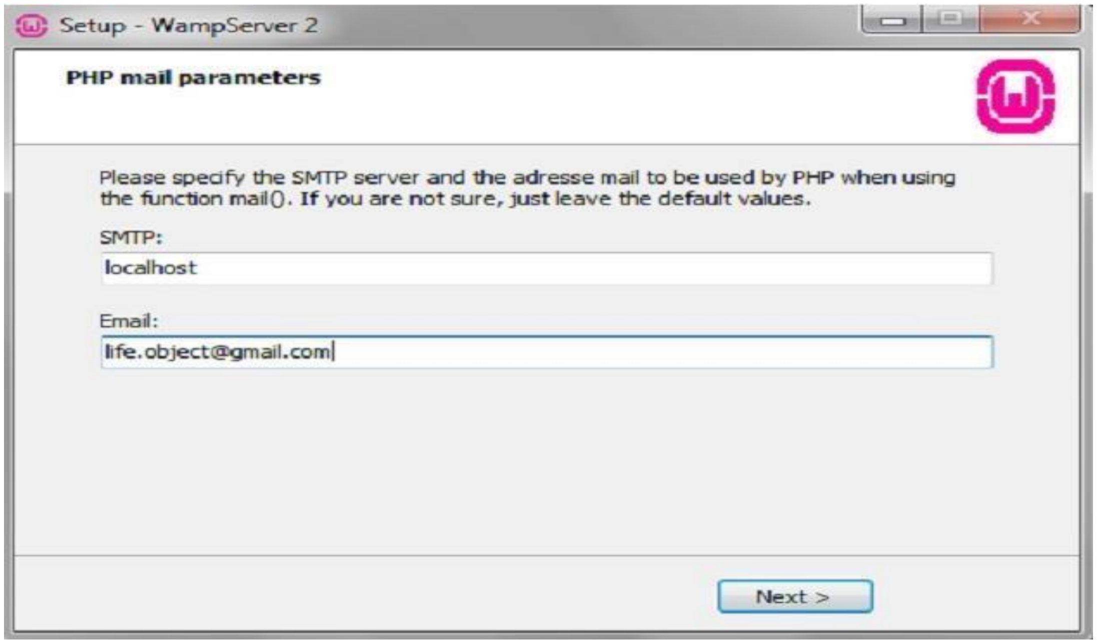

Setup - WampServer 2
PHP mail parameters
SMTP:
localhost
Email:
you@yourdomain
Windows Security Alert
Windows Firewall has blocked some features of this program
Windows Firewall has blocked some features of Apache HTTP Server on all public and private networks.
Name: Apache HTTP Server
Publisher: Apache Software Foundation
Path: D:\wamp\bin\apache\apache2.2.17\bin\httpd.exe
Allow Apache HTTP Server to communicate on these networks:
Private networks, such as my home or work network
Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)
What are the risks of allowing a program through a firewall?
Allow access
Cancel

Figure 1.33: Apache HTTP Server

11. After allowing access to Apache server, you are at SMTP server configuration dialog. Youcan specify the SMTP server and the address mail to be used by PHP when using thefunction mail(). I will recommend the following values,

- SMTP: localhost
- Email: Your email address.

Click Next after putting the above values for the installation. fiinal dialog.(Figure 1.34)



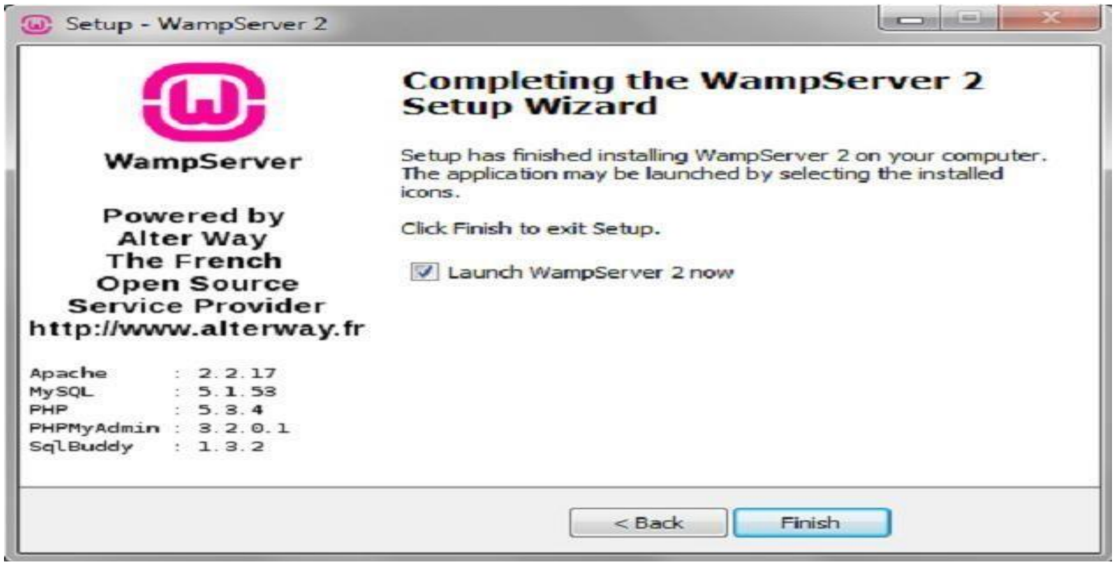
PHP mail parameters

Please specify the SMTP server and the adresse mail to be used by PHP when using the function mail(). If you are not sure, just leave the default values.

SMTP: localhost

Email: life.object@gmail.com

Figure 1.34: PHP Mail Parameters



Completing the WampServer 2 Setup Wizard

Setup has finished installing WampServer 2 on your computer. The application may be launched by selecting the installed icons.

Click Finish to exit Setup.

[✓] Launch WampServer 2 now

WampServer

Powered by Alter Way The French Open Source Service Provider http://www.alterway.fr

```
Apache      :  2.2.17
MySQL       :  5.1.53
PHP         :  5.3.4
PHPMyAdmin  :  3.2.0.1
SqlBuddy    :  1.3.2
```

Figure 1.35: WampServer 2 Setup Wizard Completion

12. You have successfully installed WampServer 2.1 d along with Apache, MySql, PHP, phpMyAdmin and SqlBuddy at your computer.

Click –Finish" tostartWampServer along with other services. Leave –Launch WampServer 2 now" check-box checked to start WampServer automatically after installation.(Fig1.35)

Web Servlets:

Servlets and JSP:

Servlet: Lifecycle of a Servlet, a simple Servlet, the servlet API, the Javax.servlet package, reading Servlet parameters, the javax.servlet. HTTP package, Handling HTTP requests and responses, using cookies and sessions.

JSP: The anatomy of a JSP page, JSP processing, declarations, directives, expressions, code snippets, implicit objects, using beans in JSP pages, connecting to database in JSP.

Servlets:

Java Servlets are programs that run on a Web or Application server and act as a middle layer between a request coming from a Web browser or other HTTP client and databases or applications on the HTTP server.

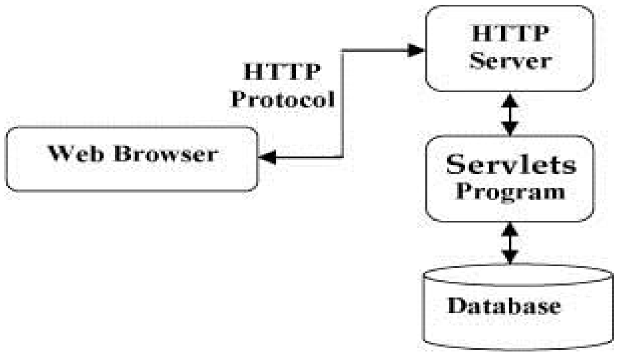
Using Servlets, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.

Java Servlets often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But Servlets offer several advantages in comparison with the CGI.

- Performance is significantlybetter.
- Servlets execute within the address space of a Web server. It is not necessary to create a separate process to handle each clientrequest.
- Servlets are platform-independent because they are written inJava.
- Java security manager on the server enforces a set of restrictions to protect the resources on a server machine. So servlets aretrusted.
- The full functionality of the Java class libraries is available to a servlet. It can communicate with applets, databases, or other software via the sockets and RMI mechanisms that you have seenalready.

Servlets Architecture:

Following diagram shows the position of Servelts in a Web Application.



Servlets Tasks:

Servlets perform the following major tasks:

- Read the explicit data sent by the clients (browsers). This includes an HTML form on a Web page or it could also come from an applet or a custom HTTP clientprogram.
- Read the implicit HTTP request data sent by the clients (browsers). This includes cookies, media types and compression schemes the browser understands, and soforth.
- Process the data and generate the results. This process may require talking to a database, executing an RMI or CORBA call, invoking a Web service, or computing the response directly.
- Send the explicit data (i.e., the document) to the clients (browsers). This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), Excel, etc.
- Send the implicit HTTP response to the clients (browsers). This includes telling the browsers or other clients what type of document is being returned (e.g., HTML), setting cookies and caching parameters, and other suchtasks.

Servlets Packages:

Java Servlets are Java classes run by a web server that has an interpreter that supports the Java Servlet specification.

Servlets can be created using the **javax.servlet** and **javax.servlet.http** packages, which are a standard part of the Java's enterprise edition, an expanded version of the Java class library that supports large-scale development projects.

These classes implement the Java Servlet and JSP specifications. At the time of writing this tutorial, the versions are Java Servlet 2.5 and JSP 2.1.

Java servlets have been created and compiled just like any other Java class. After you install the servlet packages and add them to your computer's Classpath, you can compile servlets with the JDK's Java compiler or any other current compiler.

Life cycle ofservlet

A Servlet life cycle can be defined as the entire process from its creation till the destruction. The following are the paths followed by a servlet

- The servlet is initialized by calling the **init ()** method.
- The servlet calls **service()** method to process a client's request. The servlet is terminated by calling the **destroy()** method.
- Finally, servlet is garbage collected by the garbage collector of

the JVM. Now let us discuss the life cycle methods in details.

The init() method:

The init method is designed to be called only once. It is called when the servlet is first created,and notcalledagainforeachuserrequest.So,itisusedforone-timeinitializations,justaswiththeinit method ofapplets.

The servlet is normally created when a user first invokes a URL corresponding to the servlet, but you can also specify that the servlet be loaded when the server is first started.

When a user invokes a servlet, a single instance of each servlet gets created, with each user request resulting in a new thread that is handed off to doGet or doPost as appropriate. The init() method simply creates or loads some data that will be used throughout the life of the servlet.

The init method definition looks like this:

```
publicvoid init()throwsServletException{
// Initialization code...
}
```

The service() method:

The service() method is the main method to perform the actual task. The servlet container (i.e. web server) calls the service() method to handle requests coming from the client(browsers) and to write the formatted response back to the client.

Each time the server receives a request for a servlet, the server spawns a new thread and calls service. The service() method checks the HTTP request type (GET, POST, PUT, DELETE, etc.) and calls doGet, doPost, doPut, doDelete, etc. methods as appropriate.

Here is the signature of this method:

```
publicvoid service(ServletRequest request, ServletResponse response) throwsServletException,IOException{
}
```

The service () method is called by the container and service method invokes doGe, doPost, doPut, doDelete, etc. methods as appropriate. So you have nothing to do with service() method but you override either doGet() or doPost() depending on what type of request you receive from the client.

The doGet() and doPost() are most frequently used methods with in each service request. Here is the signature of these two methods.

The doGet() Method

A GET request results from a normal request for a URL or from an HTML form that has no METHOD specified and it should be handled by doGet() method.

```
publicvoid doGet(HttpServletRequest request, HttpServletResponse response)
throwsServletException,IOException{
// Servlet code
}
```

The doPost() Method

A POST request results from an HTML form that specifically lists POST as the METHOD and it should be handled by doPost() method.

```
publicvoid doPost(HttpServletRequest request, HttpServletResponse response)
throwsServletException,IOException{
// Servlet code
}
```

The destroy() method:

The destroy() method is called only once at the end of the life cycle of a servlet. This method gives your servlet a chance to close database connections, halt background threads, write cookie lists or hit counts to disk, and perform other such cleanup activities.

After the destroy() method is called, the servlet object is marked for garbage collection. The destroy method definition looks like this:

```
publicvoid destroy(){
// Finalization code...
}
```

Servlet Deployment:

By default, a servlet application is located at the path <Tomcat-installation- directory>/webapps/ROOT and the class file would reside in <Tomcat-installation- directory>/webapps/ROOT/WEB-INF/classes.

If you have a fully qualified class name of **com.myorg.MyServlet**, then this servlet class must be located in WEB- INF/classes/com/myorg/MyServlet.class.

For now, let us copy HelloWorld.class into <Tomcat-installation-directory>/webapps/ROOT/WEB-INF/classes and create following entries in **web.xml** file located in <Tomcat-installation-directory>/webapps/ROOT/WEB-INF/

```
<servlet>
<servlet-name>HelloWorld</servlet-name>
<servlet-class>HelloWorld</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>HelloWorld</servlet-name>
<url-pattern>/HelloWorld</url-pattern>
</servlet-mapping>
```

Above entries to be created inside <web-app>...</web-app> tags available in web.xml file. There could be various entries in this table already available, but never mind.

You are almost done, now let us start tomcat server using <Tomcat-installation- directory>\bin\startup.bat (on windows) or <Tomcat-installation-directory>/bin/startup.sh (on Linux/Solaris etc.) and finally type **http://localhost:8080/HelloWorld** in browser's address box.

Example Program: Adding 2 numbers using servlet.

Index.html:

```
<html>
<body>
        <form action=”./add”>
                Enter 1st no: <input type=”text” name =”t1”/><br/>
                Enter 2nd no: <input type=”text” name=”t2”/><br/>
                <input type=”submit” value=”ADD”/>
        </form>
</body> </html>
```

<u>Web.xml:</u>

```
<?xml: version=”1.0”/>
  <web-app>
          <servlet>
                  <servlet-name>ex2</servlet-name>
                  <servlet-class>AddServlet</servlet-class>
          </servlet>
          <servlet-mapping>
                  <servlet-name>ex2</servlet-name>
                  <url-pattern>/add</url-pattern>
  </servlet-mapping>
  </web-app>
```

<u>AddServlet.java</u>:

```
import javax servlet.*;
import java.io.*;
public class AddServlet extends GenericServlet
{
        Public void service(ServletRequest req, ServletResponse res) throws IOException
        {
                int a = Integer.parseInt(req.getParameter(“t1”));
                int b = Integer.parseInt(req.getParameter(“t2”));
                int c=a+b;
                PrintWriter out=res.getWriter();
                Out.println(“Sum is:” +c);
        }
}
```

<u>Example Program</u>: To find out Factorial of a given number using servlet.

<u>Index.html</u>:

```
<html>
<body>
        <form action=”./fact”>
                Enter a no: <input type=”text” name=”t1”/><br/>
                <input type =”submit” value=”CLICK”/> </form></body></html>
```

Web.xml:

```
<?xml: version=”1.0”/>
 <web-app>
            <servlet>
                    <servlet-name>ex3</servlet-name>
                    <servlet-class>FactServlet</servlet-class>
            </servlet>
            <servlet-mapping>
                    <servlet-name>ex3</servlet-name>
                    <url-pattern>/fact</url-pattern>
 </servlet-mapping>
 </web-app>
```

FactServlet.java:

```
import javax servlet.*;
import java.io.*;
public class FactServlet extends GenericServlet
{
            Public void service(ServletRequest req, ServletResponse res) throws IOException
            {
                    int n=Integer.parseInt(req.getParameter(“t1”));
                    int i,f=1;
                    for(i=1;i<=n;i++)
                            f=f*I;
                    PrintWriter out=res.getWriter();
                    Out.println(“Factorial result  is:” +f);
            }
}
```

Example Program: To display servlet program table data frame.

Index.html:

```
<html>
            <body>
                    <form action=”./display”>
                            <input type=”submit” value=”Display Student Details”/>
                    </form>
            </body>
    </html>
```

Web.xml:

```
  <?xml: version="1.0"/>
    <web-app>
            <servlet>
                    <servlet-name>ex1</servlet-name>
                    <servlet-class>DisplayServlet</servlet-class>
            </servlet>
            <servlet-mapping>
                    <servlet-name>ex1</servlet-name>
                    <url-pattern>/display</url-pattern>
    </servlet-mapping>
    </web-app>
```

DisplayServlet.java:

```
    import java.sql.*;
    import javax.servlet.http*;
    import javax servlet.*;
    import java.io.*;
    public class DisplayServlet extends HttpServlet
    {
            Connection con;
            Statement st;
            public void init(ServletConfig sc)
            {
                    try
                    {
                            Class.forName("oracle.jdbc.driver.OracleDriver");
            con=DriverManager.getConnection("jdbc.oracle:thin:@localhost:1521:xe", "system","system");
                            System.out.println(con);
                    }
                    catch(Exception e)
                    {
                            e.printStackTrace();
                    }
                Public void doGet(HttpServletRequest req, HttpServletResponse res)
                {
                    try
                    {
                        res.setConnectType("text/html");
                        PrintWriter out=res.getWriter();
                        st=con.createStatement();
                        ResultSet rs=st.executeQuery("select * from student");
                        while(rs.next())
                        {
System.out.println("/nsno is:"+rs.getInt(1)+"\nSname is:"+rs.getString(2)+"\ncourse is:"+rs.getString(3)+"\n");
                        }
            }
                catch(Expection e)
    {
            e.printStackTrace();
    }
    }
    public void destroy()
    {
            try{con.close();}
            catch(Exception e)
    {
            e.printStackTrace();
    }
    }
    }
```

<u>Example Program</u>: To Register servlet program to insert student data.

<u>Index.html</u>:

```
<html>
        <body>
                <form action="./register">
                        rno<input type="text" name="t1"/><br/>
                        name<input type=”text” name=”t2”/><br/>
                        branch<input type=”text” name=”t3”/><br/>
                        <input type=”submit” value=”REGISTER”/>
                </form>
        </body>
</html>
```

Web.xml:

```
<?xml: version=”1.0”/>
<web-app>
        <servlet>
                <servlet-name>ex4</servlet-name>
                <servlet-class>RegisterServlet</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>ex4</servlet-name>
                <url-pattern>/register</url-pattern>
</servlet-mapping>
</web-app>
```

RegisterServlet.java:

```
import java.sql.*;
import javax.servlet.http*;
import javax servlet.*;
import java.io.*;
public class RegisterServlet extends GenericServlet
{
        Connection con;
        Statement st;
        public void init(ServletConfig sc)
        {
                try
                {
                        Class.forName(“oracle.jdbc.driver.OracleDriver”);
        con=DriverManager.getConnection(“jdbc.oracle:thin:@localhost:1521:xe”, “system”,”system”);
                                        }
                catch(Exception e)
```

```
                    {
                              e.printStackTrace();
                              }
                    Public void service(HttpServletRequest req, HttpServletResponse res)
                    {
                              try
                              {
                                        int r=Integer.parseInt(req.getParameter("t1"));
                                        String =req.getParameter("t2");
                                        String b=req.getParameter("t3");
                                        st=con.createStatement();
                                        int a =st.executeUpdate("insert into student values("+r+";"+n+"l"+b+")");
                                        PrintWriter out=res.getWriter();
                                        out.println(a+"Record inserted");
                              }

public void destroy()
{
          try{st.close();con.close();}
          catch(Exception e)
{
          e.printStackTrace();
}
}
}
```

Example Program: To login servlet program to validate login form.

login.html:

```
<html>
          <body>
                    <form action="./login">
                              Username:<input type="text" name="uname"/><br/>
                              Password:<input type="text" name="pwd"/><br/>
                    </form>
          </body>
</html>
```

<u>Web.xml:</u>

```
<?xml: version=”1.0”/>
 <web-app>
            <servlet>
                     <servlet-name>ex4</servlet-name>
                     <servlet-class>LoginServlet</servlet-class>
            </servlet>
            <servlet-mapping>
                     <servlet-name>ex4</servlet-name>
                     <url-pattern>/login</url-pattern>
 </servlet-mapping>

            <welcome-file-list>
                     <welcome-file>login.html</welcome-file>
            </welcome-file-list>
 </web-app>
```

<u>LoginServlet.java</u>:

```
import java.sql.*;
import javax.servlet.http*;
import javax servlet.*;
import java.io.*;
public class LoginServlet extends GenericServlet
{
          Connection con;
          Statement st;
          public void init(ServletConfig config)
          {
                   try
                   {
                            Class.forName(“oracle.jdbc.driver.OracleDriver”);
          con=DriverManager.getConnection(“jdbc.oracle:thin:@localhost:1521:xe”, “system”,”system”);
                                             }
                   catch(Exception e)
                   {
                            e.printStackTrace();
```

```
                }
        Public void service(HttpServletRequest req, HttpServletResponse res)
        {
                try
                {
                        String a=req.getParameter("uname");
                        String b=req.getParameter("pwd");
                        st=con.createStatement();
    ResultSet rs=st.executeQuery("select * from login where uname='"+a+"'and pwd='"+b+"'");
                        PrintWriter out=res.getWriter();
                        if(rs.next())
                {
                        out.println("welcome to:"+a);
                }
                else
                {
                        out.pritnln("wrong credentials");
                }
    }
    catch(Exception e)
{
    e.printStackTrace();
}
}
public void destroy()
{
    try{st.close();con.close();}
    catch(Exception e)
{
    e.printStackTrace();
}
}
}
```

JSP Overview

Java Server Pages (JSP) is a technology for developing web pages that support dynamic content which helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.

A JavaServer Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.

Using JSP, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.

JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages and sharing information between requests, pages etc.

Why Use JSP?

JavaServer Pages often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But JSP offer several advantages in comparison with the CGI.

- Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having a separate CGI files.
- JSP are always compiled before it's processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is requested.
- JavaServer Pages are built on top of the Java Servlets API, so like Servlets, JSP also has access to all the powerful Enterprise Java APIs, including JDBC, JNDI, EJB, JAXP etc.
- JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

Finally, JSP is an integral part of J2EE, a complete platform for enterprise class applications. This means that JSP can play a part in the simplest applications to the most complex and demanding.C **Advantages of JSP:**

Following is the list of other advantages of using JSP over other technologies:

- vs. Active Server Pages (ASP):The advantages of JSP are twofold. First, the dynamic part is written in Java, not Visual Basic or other MS specific language, so it is more powerful and easier to use.
- Second, it is portable to other operating systems and non-Microsoft Web servers.
- vs. Pure Servlets:It is more convenient to write (and to modify!) regular HTML than to have plenty of println statements that generate the HTML.
- vs. Server-Side Includes (SSI):SSI is really only intended for simple inclusions, not for "real" programs that use form data, make database connections, and the like.

- vs. JavaScript: JavaScript can generate HTML dynamically on the client but can hardly interact with the webserver to perform complex tasks like database access and image processing etc. vs. Static HTML:Regular HTML, of course, cannot contain dynamic information

JSP Declarations:

A declaration declares one or more variables or methods that you can use in Java code later in the JSP file. You must declare the variable or method before you use it in the JSP file. Following is the syntax of JSP Declarations:

<%!declaration;[declaration;]+...%>

You can write XML equivalent of the above syntax as follows:

<jsp:declaration> code fragment

</jsp:declaration>

Following is the simple example for JSP Declarations:

<%!int i =0;%>

<%!int a,b,c;%>

<%!Circle a =newCircle(2.0);%>

<\%Represents static <% literal.%\> Represents static %>

literal.\'A single quote in an attribute that uses single quotes.\"A double quote in an attribute that uses double quotes.

JSP Directives:

A JSP directive affects the overall structure of the servlet class. It usually has the following form:

<%@directive attribute="value"%> There are three types of directive tag: Directive

Description

<%@ page ... %> Defines page

-

dependent attributes, such as scripting language, error page, and buffering requirements.

<%@ include ... %>

Includes a file during the translation phase.

<%@ tag lib ... %>

Declares a tag library, containing custom actions, used in the page The <jsp:forward> Action

The forward action terminates the action of the current page and forwards the request to another resource such as a static page, another JSP page, or a Java Servlet.

The simple syntax of this action is as follows:

<jsp:forward page="Relative URL"/>

Following is the list of required attributes associated with forward action: Attribute

Description page

Should consist of a relative URL of another resource such as a static page, another JSP page, or a Java Servlet

Example:

Let us reuse following two files (a) date.jps and (b) main.jsp as follows:

Following is the content of date.jsp file:

```
<p>
Today's date:
<%=(new java.util.Date()).toLocaleString()%>
</p>
```

Here is the content of main.jsp file:

```
<html>
<head>
<title>
The include Action Example
</title>
</head>
<body><center>
<h2>
The include action Example
</h2>
<jsp:forward page="date.jsp"/>
</center>
</body>
</html>
```

Now let us keep all these files in root directory and try to access main.jsp. This would display r esult something like as below. Here it discarded content from main page and displayed contentfrom forwarded pageonly.

Today's date: 12-Sep-2010 14:54:22 The <jsp:plugin> Action

The plugin action is used to insert Java components into a JSP page. It dete rmines the type of browser and inserts the <object> or <embed> tags as needed.If the needed plugin is not present, it downloads the plugin and then executes the Java component. The Java component can be either an Applet or a JavaBean.

The plugin action has several attributes that correspond to common HTML tags used to format Java components.

The <param> element can also be used to send parameters to the Applet or Bean. Following is the typical syntax of using plugin action:

```
<jsp:plugin type="applet"code base="dirname"code="MyApplet.class" width="60",height="80">
```

```
<jsp:paramname="fontcolor" value="red"/>
<jsp:param name="background"value="black"/>
<jsp:fallback>
Unable to initialize Java Plugin
</jsp:fallback>
</jsp:plugin>
```

You can try this action using some applet if you are interested. A new element, the <fallback> element, can be used to specify an error string to be sent to the user in case the component fails. The <jsp:element> Action
The <jsp:attribute> Action The <jsp:body> Action
The <jsp:element>, lt;jsp:attribute> and <jsp:body> actions are used to define XML elements dynamically. The word dynamically is important, because it means that the XML elements can be generated at request time rather than statically at compile time.
Following is a simple example to define XML elements dynamically:

```
<%@page language ="java" contentType="text/html"%>
<html xmlns=http://www.w3c.org/1999/xhtml xmlns:jsp="http://java.sun.com/JSP/Page">
<head><title>
Genera
te XML Element
</title></head>
<body>
<jsp:element name="xmlElement">
<jsp:attribute name="xmlElementAttr"> Value for the attribute
</jsp:attribute>
<jsp:body>
Body for XML element
</jsp:body>
</jsp:element>
</body>
</html>
```

This would produce followin g HTML code at run time:

```
<html xmlns=http://www.w3c.org/1999/xhtml xmlns:jsp="http://java.sun.com/JSP/Page">
<head><title>
Generate XML Element
</title></head>
```

```
<body>
<xmlElement xmlElementAttr= "Value for the attribute"> Body for XML element
</xmlElement>
</body>
</html>
```

The <jsp:text> Action

The <jsp:text> action can be used to write template text in JSP pages and documents. Following is the simple syntax for this action:

```
<jsp:text> Template data
</jsp:text>
```

The body fo the template cannot contain other elements; it can only contain text and EL expressions (Note: EL expressions are explained in subsequent chapter). Note that in XML files, you cannot use expressions such as ${whatever > 0}, because the greater than signs are illegal. Instead, use the gt form, such as ${whatever gt 0} or an alternative is to embed the value in a CDATA section.

* Introduction to JSP :-

JSP - Java Server page.

→ JSP is a Technology based on The java language and enables the development of dynamic web sites.

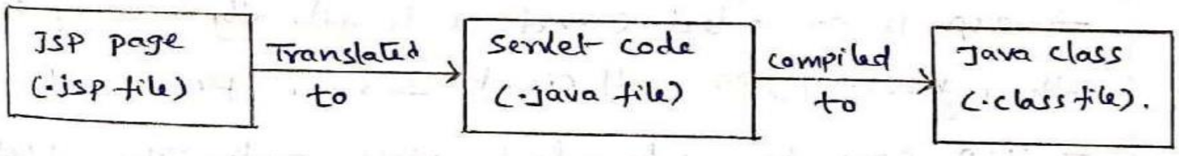
→ JSP was developed by sun micro systems to allow us to server side development.

→ A JSP page is a textual document that describes how to create a response object from request object for given protocol.

→ The main goal of JSP technology is to simplify The Certain and maintenace of server side html pages.

→ A JSP page contains standard markup elements such as HTML tags just like a regular web pages. And it contains special JSP elements that allows the server to insert dynamic content in The page.

The phases of a JSP page



* The problem With servlet :-

programming with servlets is a very boring and continuing for long time, as a servlets not only handle the request processing and bussiness logic but also the presentation task.

The servlet has some problems as described below

→ It is a common fact that servlets incorporate both the processing logic and presentation logic.

→ In the servlet, every single change in the presentation logic of an application, requires the servlet programmer to update the servlet code and recompile it.

→ Web page development tools do not directly supports servlets when designing application.

Due to these problems, sun microsystems developed JSP Technology.

→ With JSP Technology the presentation logic may contain both static and dynamic content thereby making the web development very easy and flexible.

→ In JSP Technology, the business logic and request process logic are separated from presentation logic.

→ JSP provides a very powerful and flexible mechanism to provide dynamic web pages.

→ JSP doesn't require any special setup at the client-side.

→ JSP provides built in supports for HTTP session management.

→ JSP is compiled (and automatically recompiled when nessory) for efficient server processing.

→ JSP can be integrated with enterprise Java API.

- → JDBC API
- → Thread API
- → EJB API

→ Web development tools gives full support to JSP because JSP pages are like HTML pages.

* In linux (ubuto), JSP files will be saved in

var - lib - tomcat 7 - webapps

* To run localhost:8080 / foldername / file name (JSP).

port no of Tomcat to servlets & JSP files

* The Anatomy of JSP page :-

The JSP page is almost like a regular web page with the inclusion of dynamic behaviour through JSP elements.

Hence we can say that a JSP page has two components.

⇒ JSP Elements :- These are the things that the JSP container understands and translates. Hence we can say that JSP elements instructs the JSP container, what code to generate and how it should operate.

⇒ Template Data :- Everything else in the page that the JSP container doesn't understand is called template data.

Actually these are the static portion of the JSP page which are passed through the JSP container unprocessed to the browser.

Example:-

```
<%@ page import="java.util.Date" session="false" %>
<html>
<head>
<title> simple JSP page </title>
</head>
<body>
<h3> current time is : </h3>
<%=new Date() %>
</body>
</html>
```

Note:- The JSP container after processing the JSP page, merges both, the static data (template data) and dynamic data, generated by processing the JSP elements. And finally the resultant content is sent to the browser as a response.

↳ <u>JSP Elements</u>:-

All the Jsp elements may be written in one of two forms.

→ The XML form

→ The <%...%> form

These are four types of JSP elements, They are

1) Directive Elements 2) Scripting Elements

3) Action Elements 4) Expression language expressions.

⇒ 1) <u>Directive Elements</u>:-

Directives gives message to JSP container, during translation to generate the corresponding servlet and to control some characteristics of the JSP page.

Directives have the following general form

$$\boxed{\underbrace{\text{<%@ directive}}_{\text{Name of directive}}\ \underbrace{\text{[attribute}_1\text{ = value}_1\text{, ...]}}_{\text{optional}}\text{ %>}}$$

These are three standard directives they are

(a) page (b) include (c) taglib.

- (a) <u>page</u>:- It is used to specify current page settings as a whole.

<u>Syntax</u>:- $\boxed{\text{<%@page [attribute}_1\text{ = "value}_1\text{" ...]%>}}$

It defines following attributes

(i) import (ii) contentType (iii) session (iv) buffer.

- (b) <u>include</u>:- It is used to include content from other resources during the translation time.

<u>Syntax</u>:- $\boxed{\text{<%@include file="url" %>}}$

(c) taglib: [illegible] functionality by making use of custom actions defined in the tag library.

Syntax :-
```
<%@ taglib uri=" libURI" prefix= "name" %>
```

Example :- "Directive. JSP"
```
<%@page import = "java.util.Date"%>
<html>
<body>
<% Date date = new Date ();
     System.out.println ("Server time is now:");
     System.out.println (date);
 %>
  Going to include welcome.html file <br/>
 <%@ include file=" welcome.html"%>
 </body>
 </html>
```

welcome.html
```
<html>
<body>
<h2> Welcome to JSP programming </h2>
<h3> Thank you </h3>
</body>
</html>
```

⇒ 2) Scripting Elements :-

Scripting elements enable programmers to embed code in another programming language (Java) within a JSP page.

They are actually executed at request processing time and are used for a varity of purposes.

Such as

→ To manipulate objects

→ To perform calculation on runtime values.

→ To perform computation on an object.

→ To declare methods or variables.

Scripting elements are 3 types. They are

→ (a) Scriptlets :-

scriptlets are the code fragments of a language which are executed at run time to process an Http reque

Scriptlets have following form:

```
<% Java code %>   (or) <% statement1 [statement2 --] %>
```

→ (b) Declarations :-

Declaration enable a programmer to declare variables and methods of a language, which can be used from any point in the JSP page.

unlike scriptlets They cannot access JSP implict objects

Declarations have following form:

```
<%! statement1 [statement2 - - ] %>
```

→ (c) Expressions :-

Expressions enables a programmer to write expres in a language which gets evaluated at request processi time. These are used to insert values directly into the output.

It has following form:

```
<%= expression %>
```

Example :- "Script.JSP"

```
<%@ page import = "Java . Util . Date" %>
<html>
<body>
<%
 System.out.println ("Evaluating date now");
```

```
Date date = new Date();
%>
Hello! The time is now
<% out.println (date); %>
<%! Date d = new Date();
    Date getDate()
    {
     return d;
    }
 %>
 Hello ! The time is now <% = getDate()%>
</body>
</html>
```

⇒ 3) Action Elements :-

Specific functionality is encapsulated by action elements in predefined tags, to be used by programmer in JSP page.

At translation time these actions elements are replaced by Java code which corresponds to specify functionality.

The action elements can dynamically generate HTML.

The action Elements should be represented using strictly XML syntax as shown below

```
<prefix: action  [attribute, = "value; ---] >
         ↓                 ↓
      TagName          optional
  //body
</prefix : action>
```

There are standard actons all are have 'jsp' as their prefix. Hence, they have the following Syntax.

```
<jsp: tagname [attribute, = "value,"... ]>
   // body
<jsp: tagname >
```

The Jsp container supports the following standard action Elements

- <jsp: include> – Dynamically includes the content other resources at request process
- <jsp: forward> – Terminates the current jsp page execution, and forwards the Http request to another Jsp for processing
- <jsp: useBean> – specifies that a JavaBean instance is used by the Jsp page.
- <Jsp: Element> – Dynamically generates an XML elements.
- <jsp: body> – Specifies the body of a tag.
- <Jsp: text> – Encloses template data.

Example :-

```
<%@ page import = "Java. util. *" %>
<html>
<body>
Here including the action
<jsp: include page= "abc. html" />
Here forwarding the page
<jsp: forward page = "xyz. jsp" />
</body>
</html>
```

* JSP processing :-

Any webserver needs a container to run any web component (or) to provide interface to run web components.

A JSP page needs a JSP container to be processed.

Generally speaking a JSP container acts as a mediator between the JSP page and the server by taking all JSP requests and to produce the response

$$\begin{tikzcd}
\boxed{\text{JSP page (.jsp file)}} \arrow[r, "\text{Translated to}"] & \boxed{\text{Servlet code (.java file)}} \arrow[r, "\text{compiled to}"] & \boxed{\text{Java class file (.class file).}}
\end{tikzcd}$$

⇒ Converting JSP page to a servlet (.java file) and compiling servlet is called "translation phase".

⇒ The JSP container is also responsible for invoking JSP page implementation to process each request and generate the response. i.e converting the servlet code to .class file is called as "Request processing phase".

Note :-

All JSP files converted to Java, that are stored in below path :

" C:\ program Files \ Apache Software Foundation \ Java \ work \ catalina \ Localhost \ Jsbean \ org \ apache \ JSP."

*

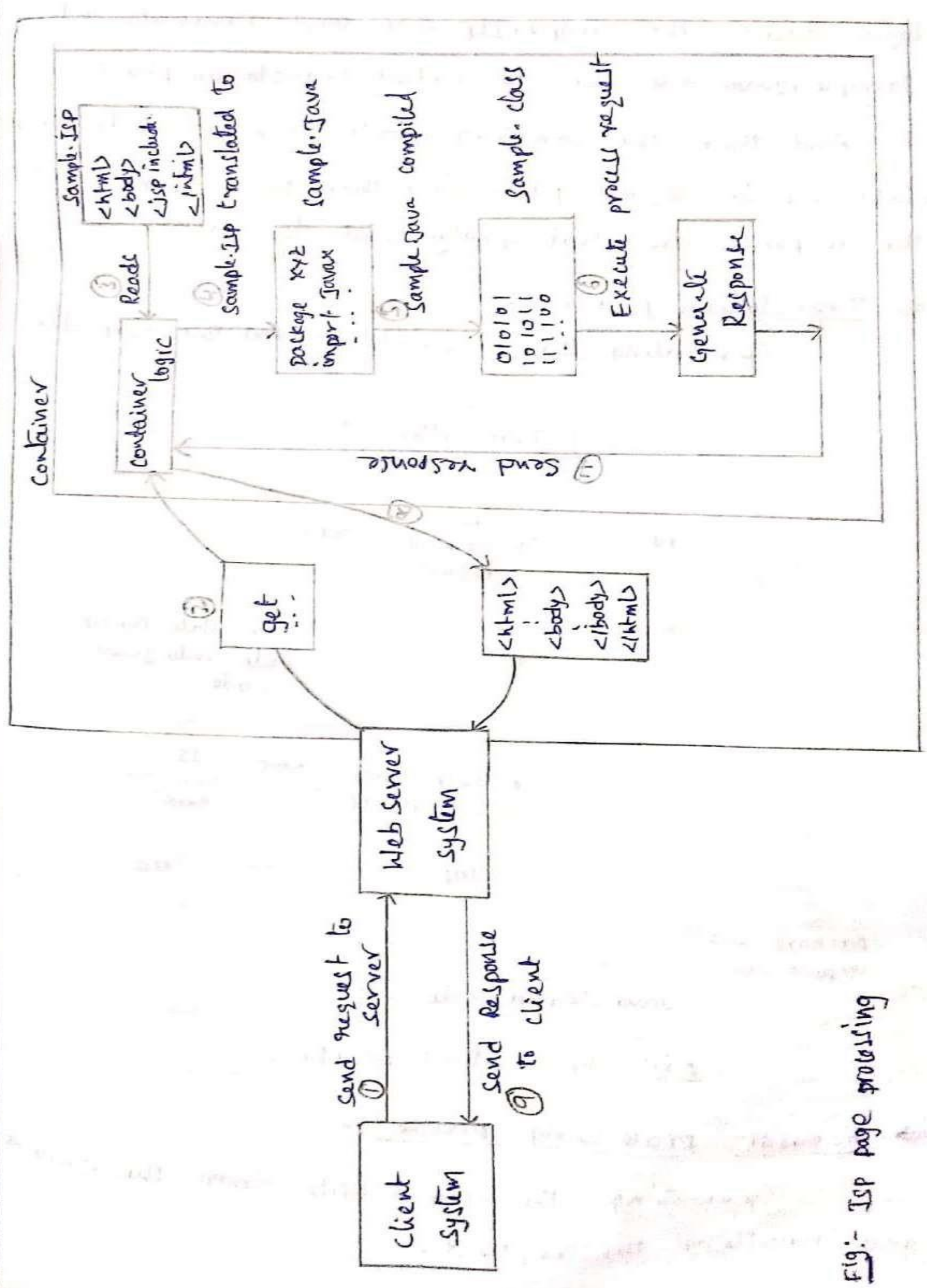


Fig:- JSP page processing

As shown in figure, the client sends the request to server, the server have container in that the container logic reads the sample.JSP file and converts into sample.Java file. This is called translation phase.

And then, the sample.Java file converted into sampl class file to request processing, then the container gener the response and that finally send to client.

⇒ Translation phase :-

Generating the .Java file from the .jsp file.

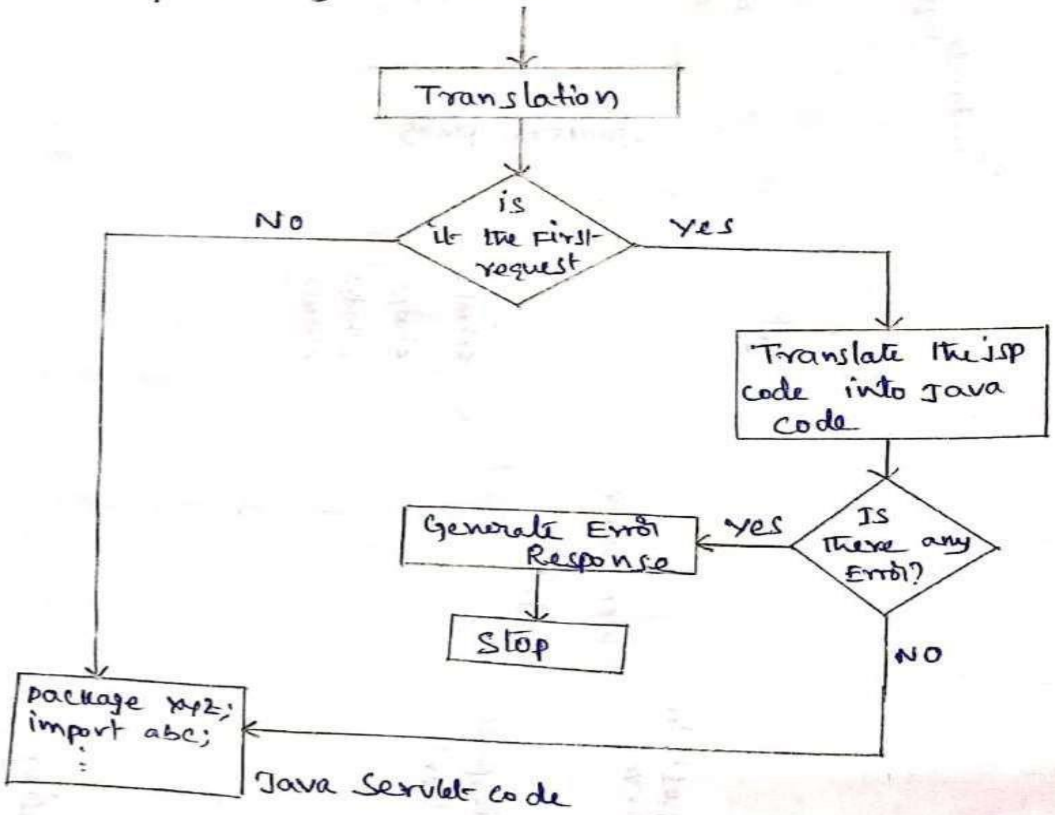


Fig:- Translation phase.

⇒ Request processing phase :-

Generating the .class file from the .Java fil and handling the request.

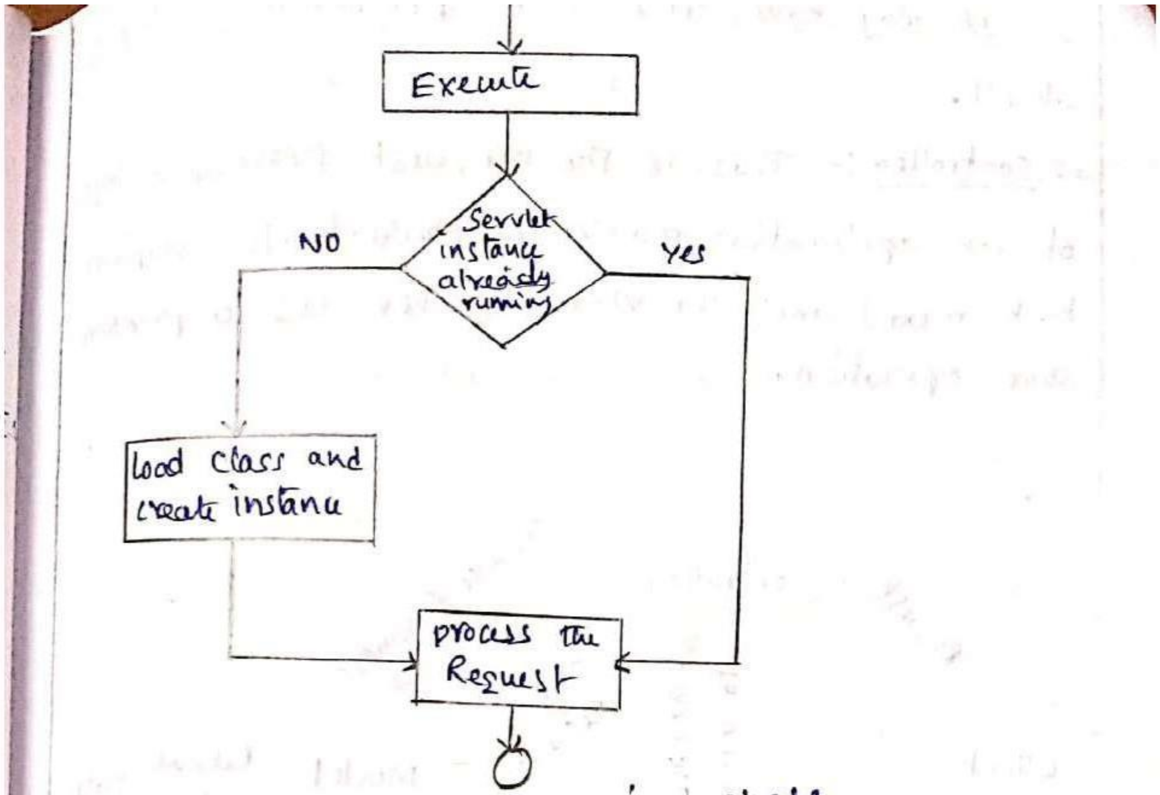

Execute
Servlet instance already running
NO
Yes
load class and create instance
process the Request

Fig:- Request processing phase.