JAVA SERVLETS

4KWJ002 - LECTURE 1

Java Servlets

By
Sudha Agarwal
sudha.agarwal@4kitsolutions.com

http://www.4kitsolutions.com

INDEX

INTRODUCTION TO WEB

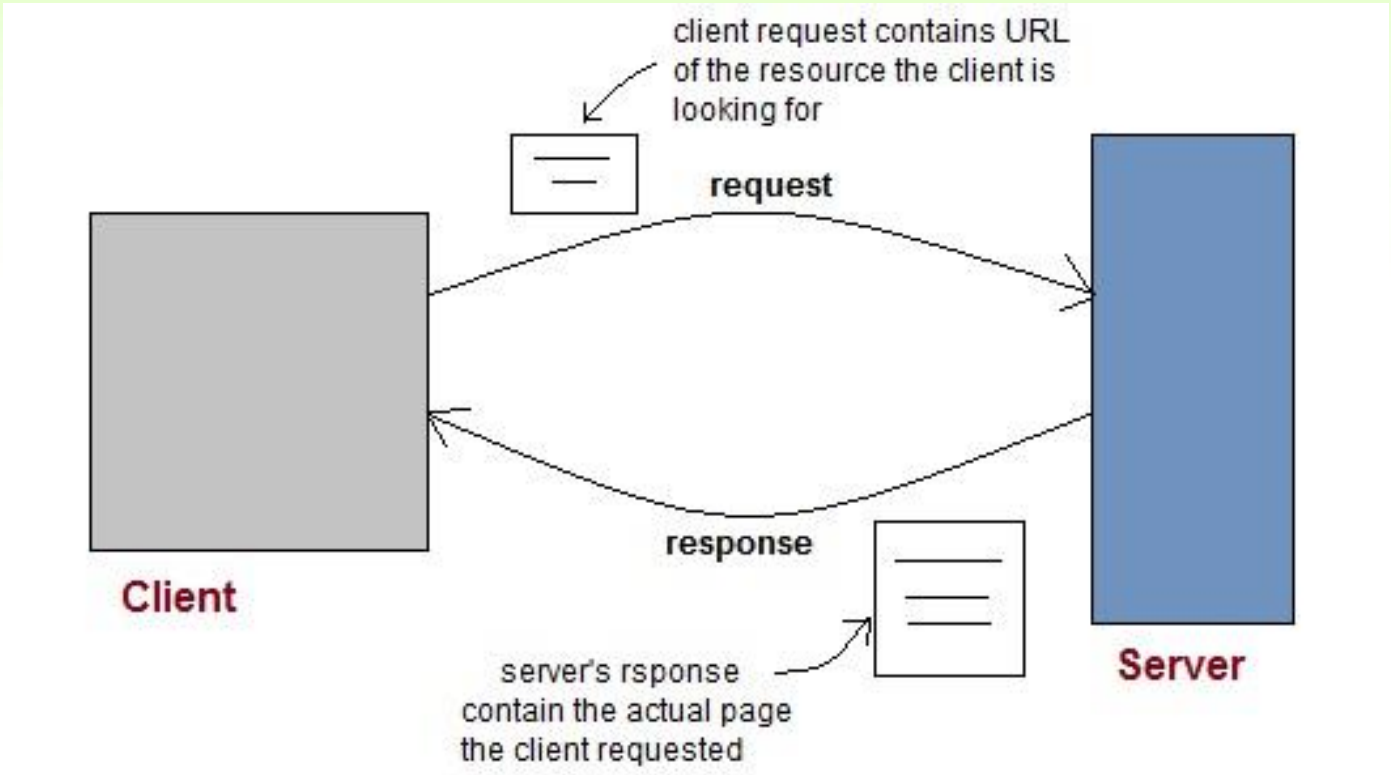
❖Web consists of billions of clients and server connected through wires and wireless networks.

❖The web clients make requests to web server. The web server receives the request, finds the resources and return the response to the client. When a server answers a request, it usually sends some type of content to the client.

❖The client uses web browser to send request to the server. The server often sends response to the browser with a set of instructions written in **HTML**(HyperText Markup Language).

❖All browsers know how to display HTML page to the client.

INTRODUCTION TO WEB



Web Application

A website is a collection of static files(webpages) such as HTML pages, images, graphics etc.

A Web application is a web site with dynamic functionality on the server. Google, Facebook, Twitter are examples of web applications.

HYPERTEXT TRANSFER PROTOCOL

❖**HTTP is a protocol that clients and servers use on the web to communicate.**

❖It is similar to other internet protocols such as SMTP(Simple Mail Transfer Protocol) and FTP(File Transfer Protocol) but there is one fundamental difference.

❖HTTP is a **stateless protocol** i.e. HTTP supports only one request per connection. This means that with HTTP the clients connect to the server to send one request and then disconnects. This mechanism allows more users to connect to a given server over a period of time.

❖The client sends an HTTP request and the server answers with an HTML page to the client, using HTTP.

HTTP METHODS

❖HTTP request can be made using a variety of methods, but the ones you will use most often are Get and Post.

❖The method name tells the server the kind of request that is being made, and how the rest of the message will be formatted.

HTTP METHODS

Method Name	Description
OPTIONS	Request for communication options that are available on the request/response chain.
GET	Request to retrieve information from server using a given URI.
HEAD	Identical to GET except that it does not return a message-body, only the headers and status line.
POST	Request for server to accept the entity enclosed in the body of HTTP method.
DELETE	Request for the Server to delete the resource.
CONNECT	Reserved for use with a proxy that can switch to being a tunnel.
PUT	This is same as POST, but POST is used to create, PUT can be used to create as well as update. It replaces all current representations of the target resource with the uploaded content.

GET VS. POST

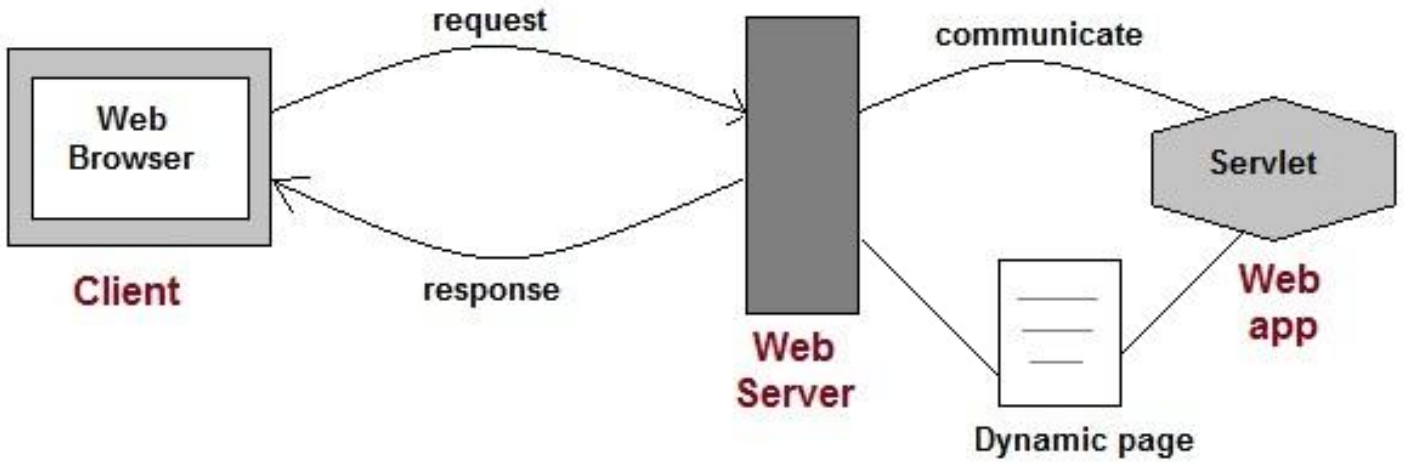
GET Request	POST Request
Data is sent in header to the server	Data is sent in the request body
Get request can send only limited amount of data	Large amount of data can be sent.
Get request is not secured because data is exposed in URL	Post request is secured because data is not exposed in URL.
Get request can be bookmarked and is more efficient.	Post request cannot be bookmarked.

INTRODUCTION TO SERVLET

- Servlet technology uses **Java language** to create web applications.
- Servlet is a **server side** programming language.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is **an interface** that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.

SERVLET WORKFLOW

❖Web applications are helper applications that resides at web server and build dynamic web pages. A dynamic page could be anything like a page that randomly chooses picture to display or even a page that displays the current time.



❖As Servlet Technology uses Java, web applications made using Servlet are **Secured, Scalable and Robust.**

SERVLET API

Servlet API consists of **two important packages** that encapsulates all the important classes and interface, namely :-

- **javax.servlet**
- **javax.servlet.http**

- The **javax.servlet** package contains many interfaces and classes that are used by the servlet or web container. These are not specific to any protocol.

- The **javax.servlet.http** package contains interfaces and classes that are responsible for http requests only.

SERVLET API...

GenericServlet class

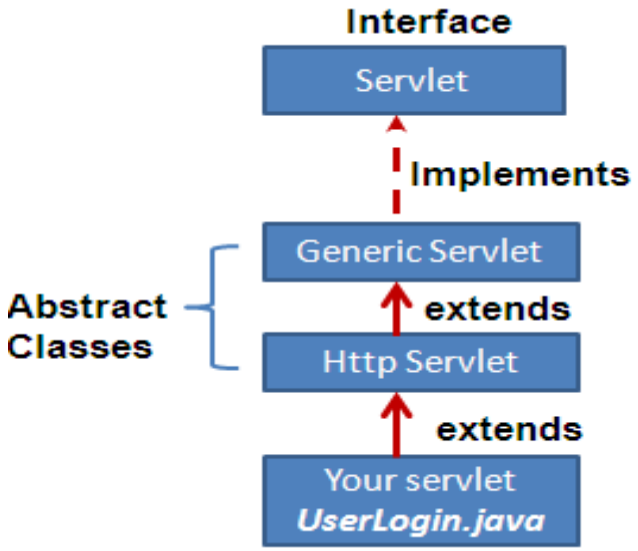
implements **Servlet**, **ServletConfig** and **Serializable** interface. It provides the implementation of all the methods of these interfaces except the service method.

This class can handle any type of request so it is protocol-independent.

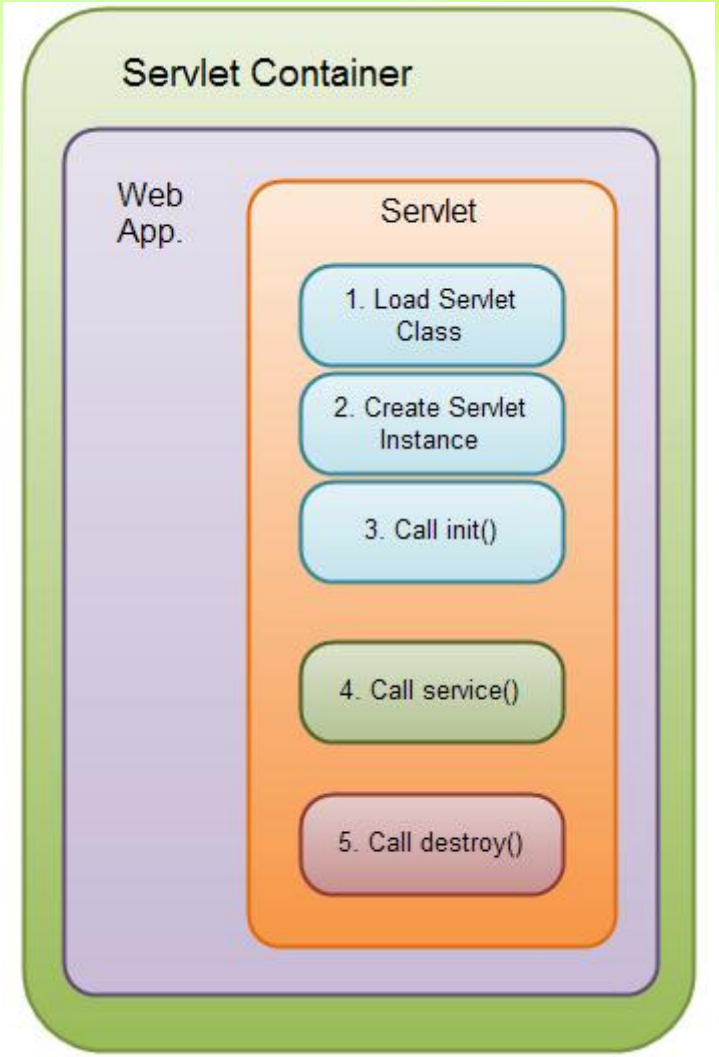
HttpServlet

This class extends the GenericServlet class and implements Serializable interface.

It provides http specific methods such as doGet, doPost, doHead, doTrace etc.



SERVLET LIFECYCLE



SERVLET LIFECYCLE - STEPS

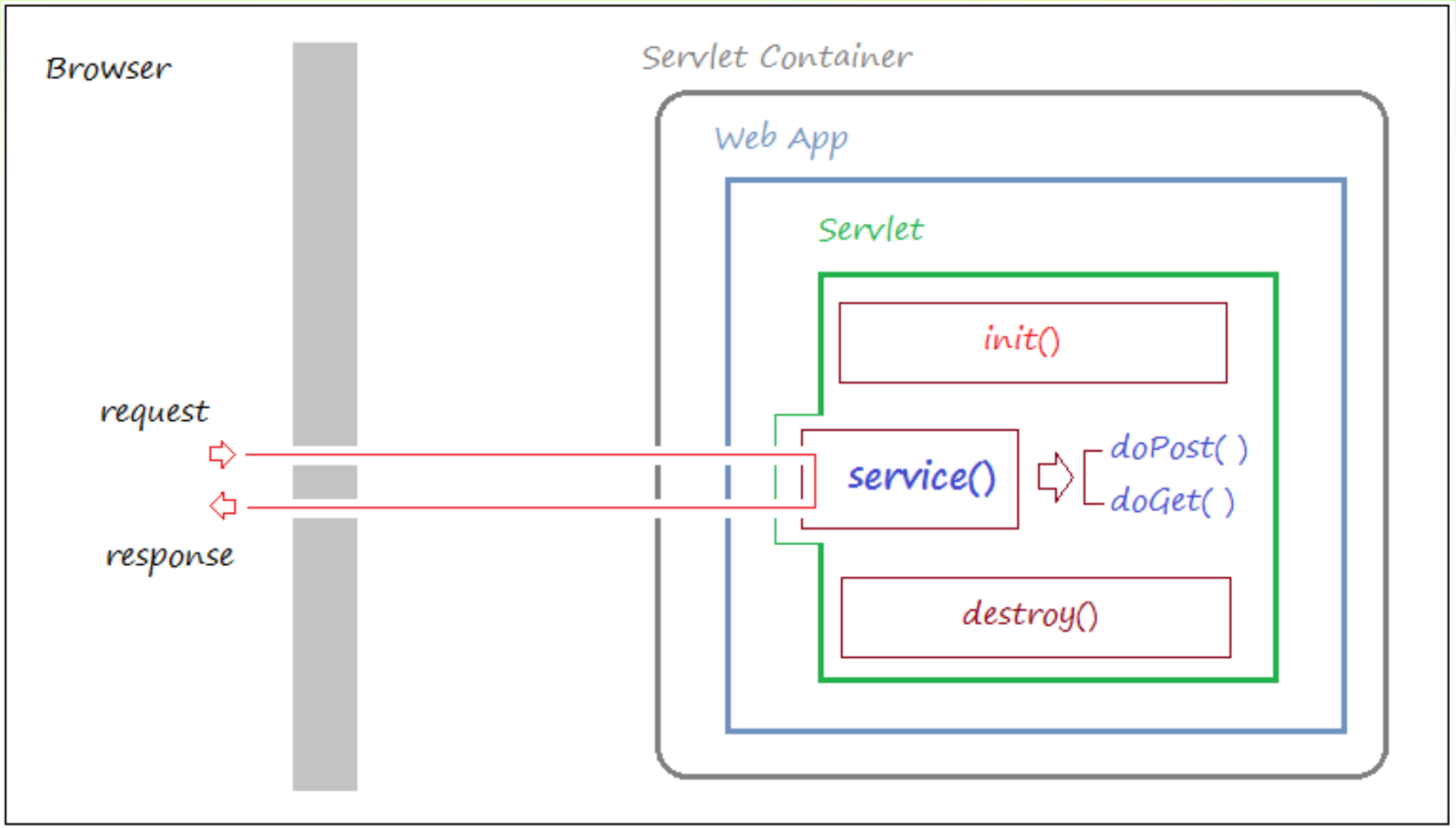
1. Load **Servlet** Class.
2. Create Instance of **Servlet**.
3. Call the servlets **init()** method.
4. Call the servlets **service()** method.
5. Call the servlets **destroy()** method.

❖Step 1, 2 and 3 are executed only once, when the servlet is initially loaded. By default the servlet is not loaded until the first request is received for it. You can force the container to load the servlet when the container starts up though.

❖Step 4 is executed multiple times - once for every HTTP request to the servlet.

❖Step 5 is executed when the servlet container unloads the servlet.

SERVLET LIFECYCLE



SERVLET LIFECYCLE

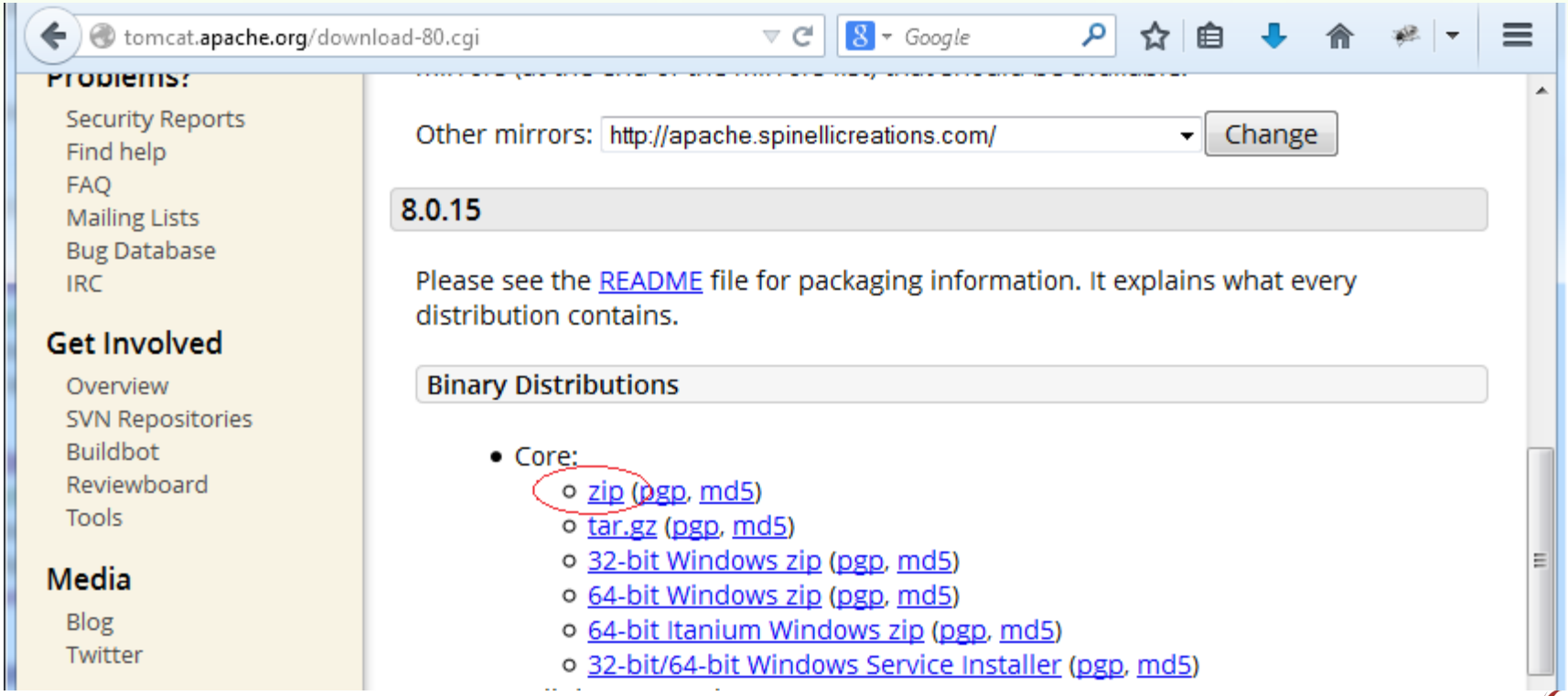
❖When the request of users to Servlet, the servlet will call the method of service() to serve the requirements of users, the service() will call either doGet() or doPost().

❖Thus, when users request a Servlet, the servlet will be created at the time of the first request, and will simultaneously call the init() method of servlet to initialize it.

❖init() method is called only one time. Method of destroy() is used to destroy the servlet, it will be called only once when removing deployment (undeloy) of web application or stop the web server.

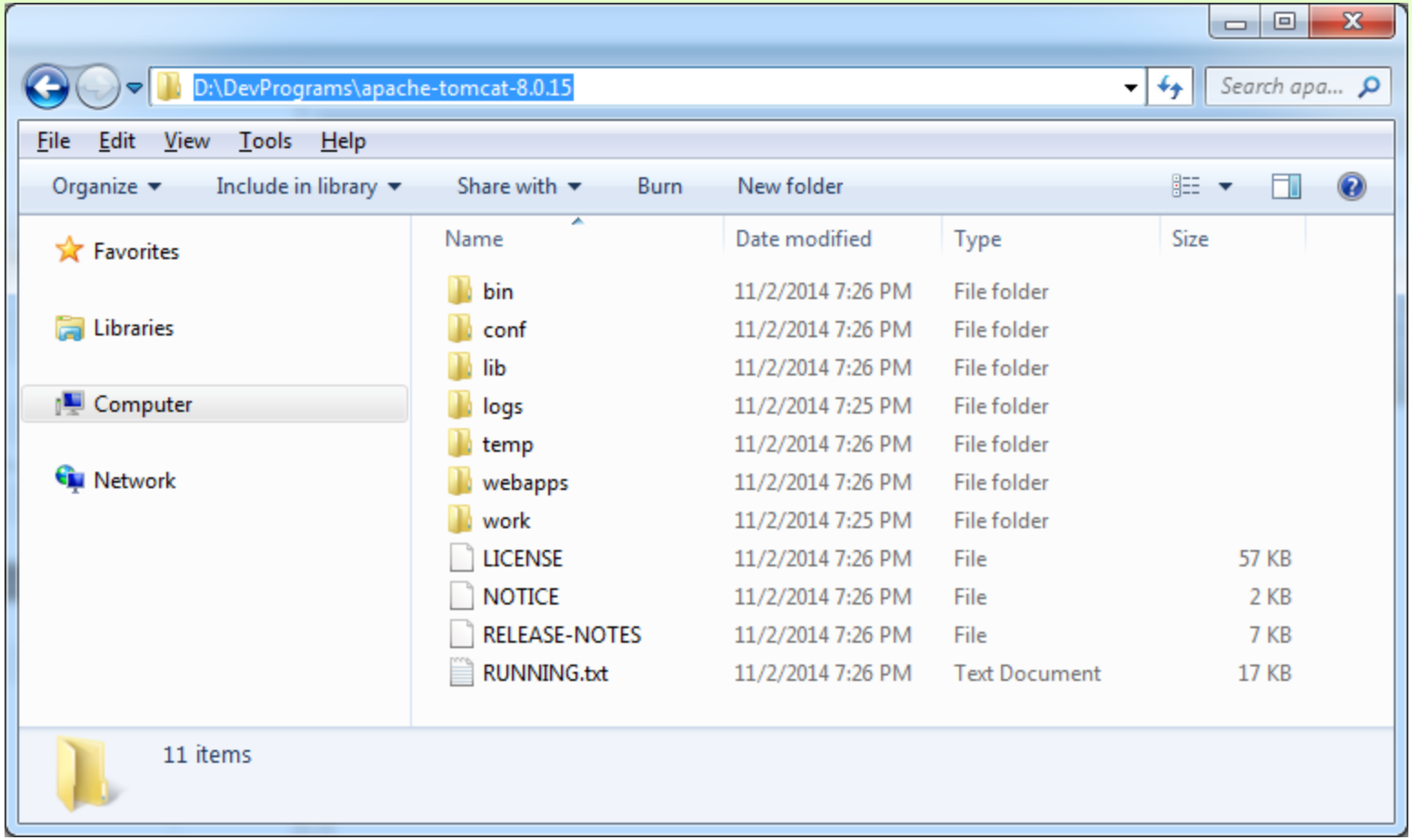
INSTALLING TOMCAT SERVER

- Download latest version of Tomcat Server from http://tomcat.apache.org/download-80.cgi
- And download the file (zip) as shown below.



INSTALLING TOMCAT SERVER

- Unzip Tomcat to a specific folder on the hard drive, such as:
- D:/DevPrograms



STARTING TOMCAT SERVER

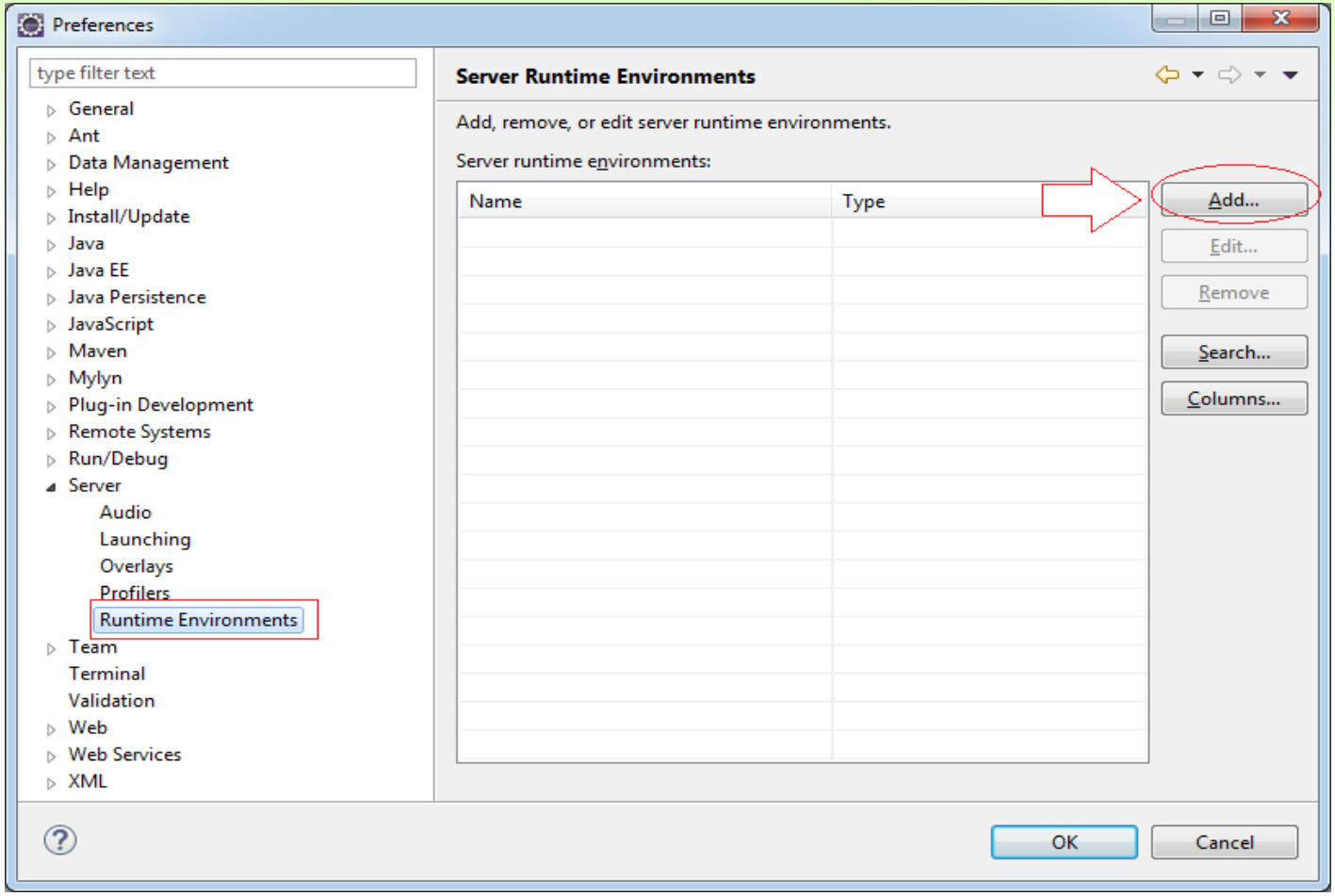
- Start a Command Prompt from the Start menu.
- Navigate to the Tomcat bin directory, e.g., c:/Tomcat8/bin :
- Type in startup and then hit Enter to execute the Tomcat server start up script:

DECLARING TOMCAT WITH ECLIPSE

- Help --> Install New Software--> place this URL in work with field
- Eclipse Oxygen: http://download.eclipse.org/releases/oxygen/
- Select WEB XML, JEE option

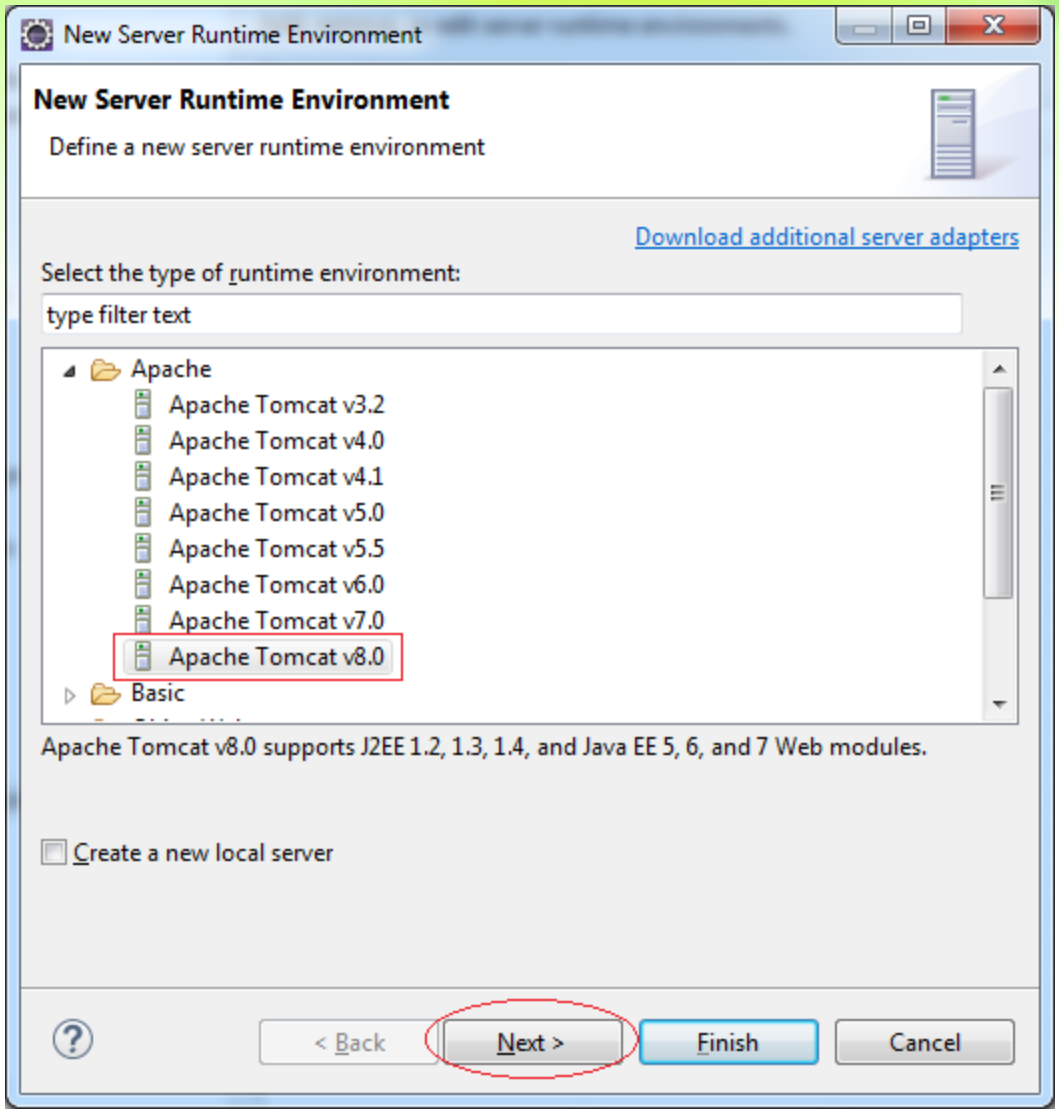
DECLARING TOMCAT WITH ECLIPSE

- Create new Runtime Server



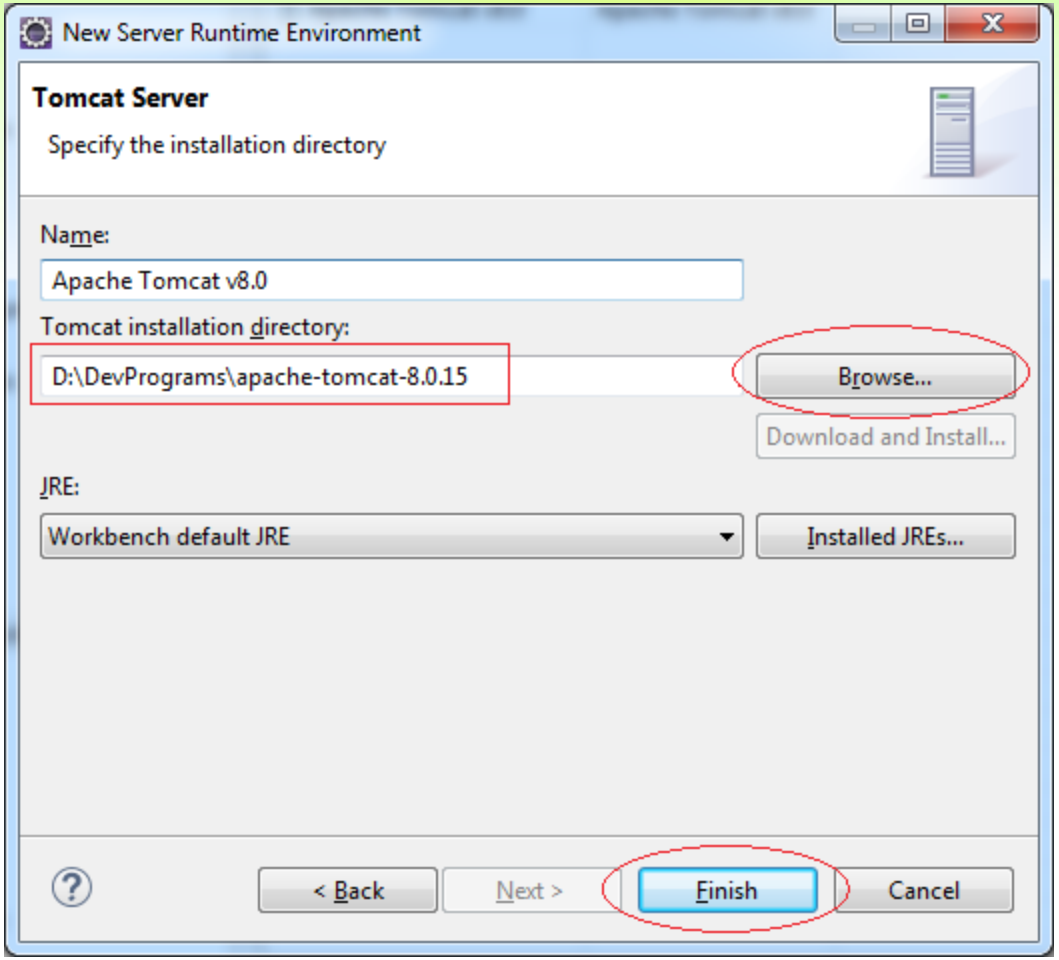
DECLARING TOMCAT WITH ECLIPSE

- Select Tomcat 8



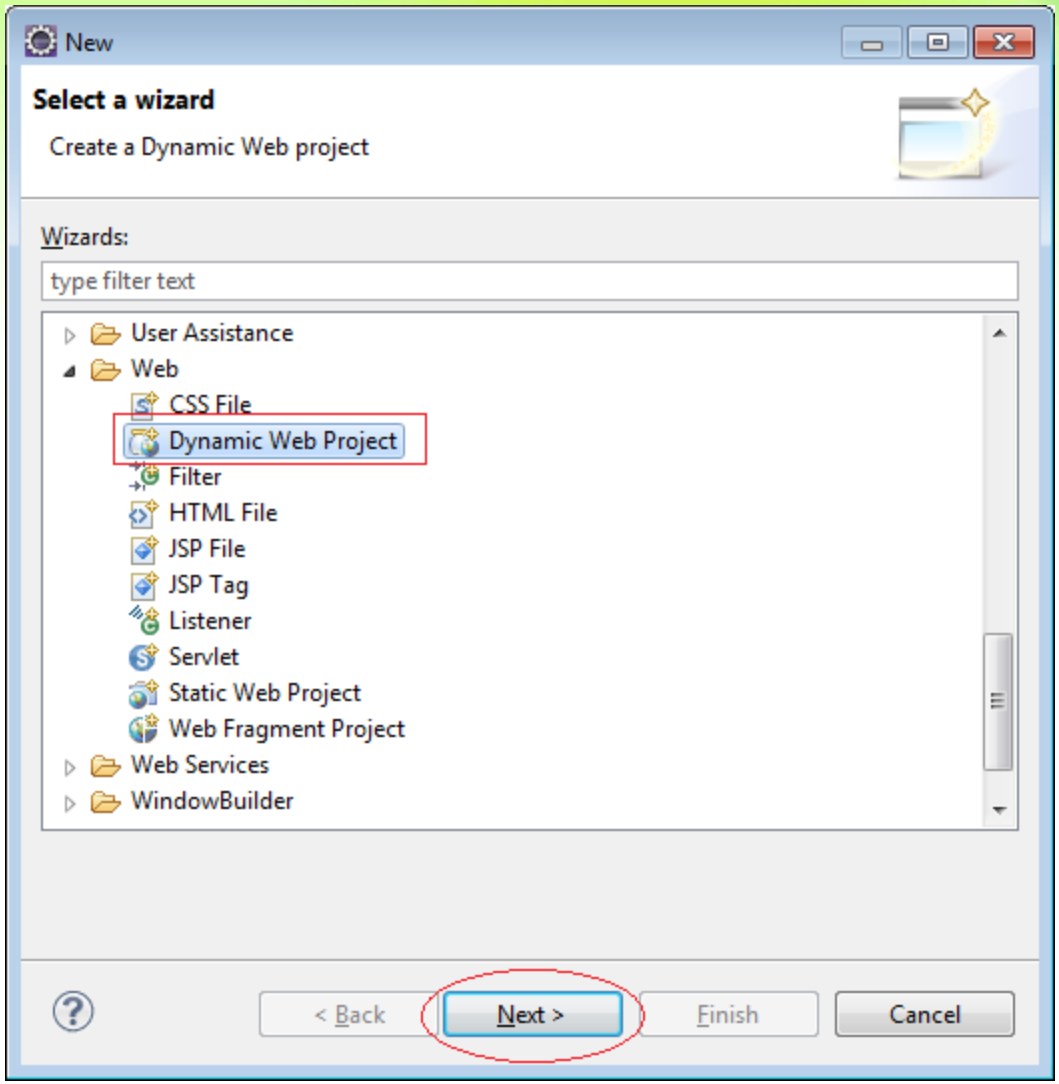
DECLARING TOMCAT WITH ECLIPSE

- Declaring location
- of Tomcat.
- Click OK to finish.



CREATING WEB PROJECT

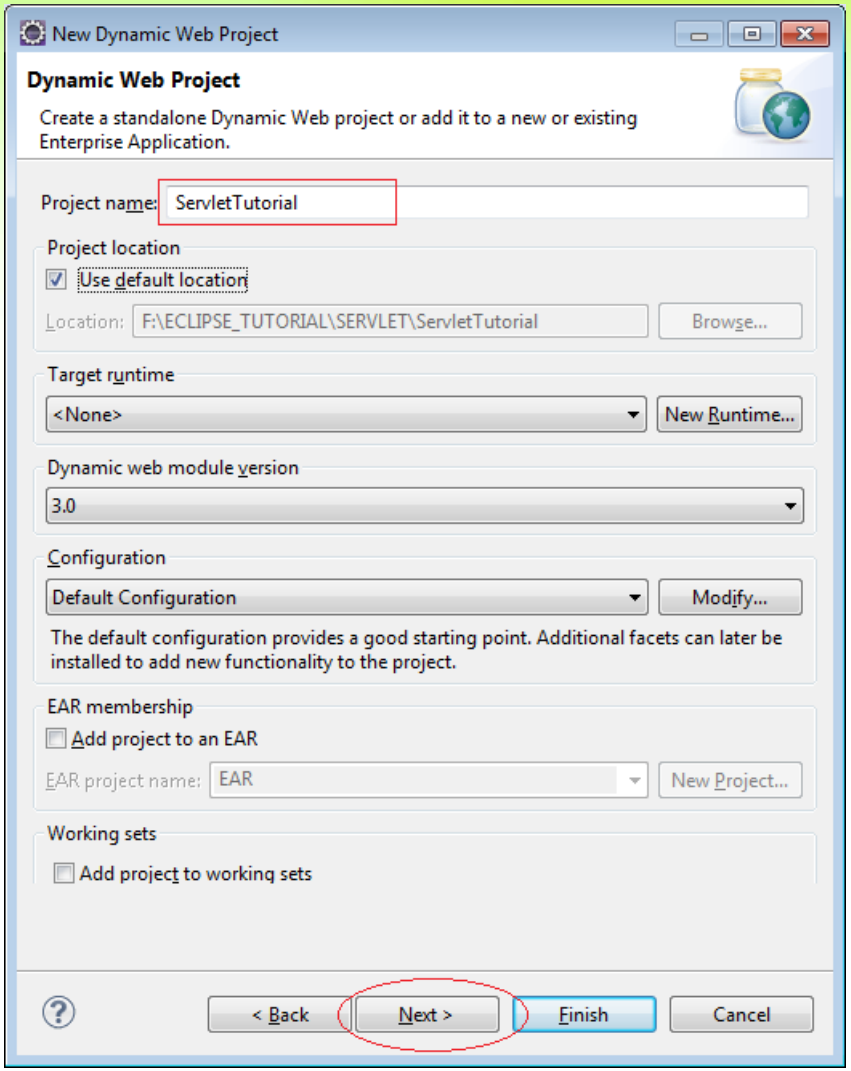
- File/New/Other



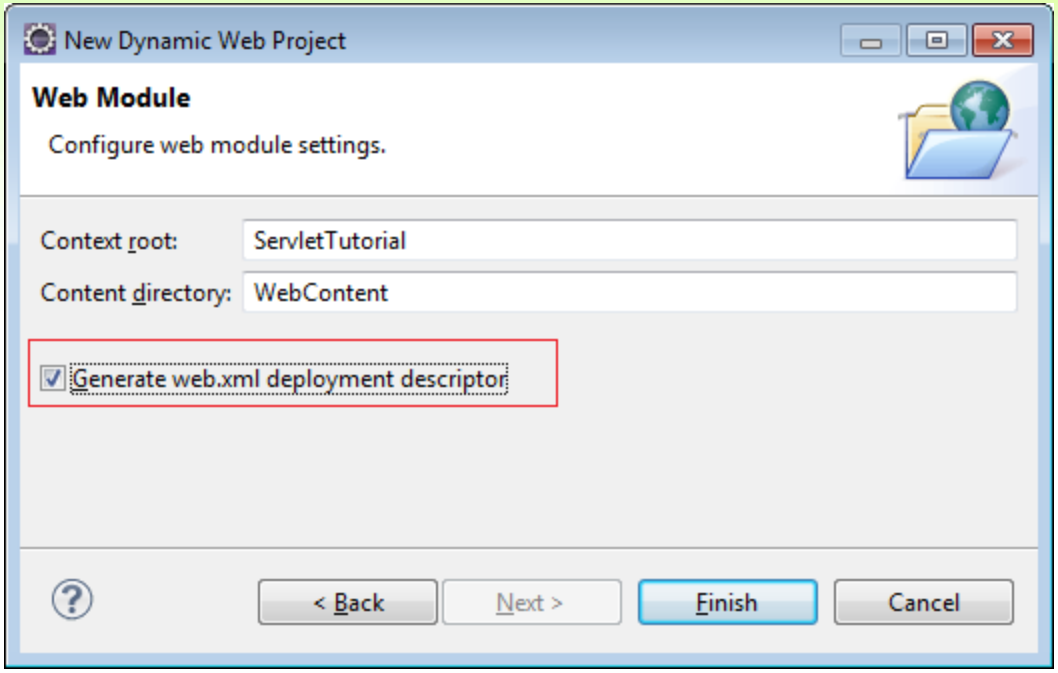
CREATING WEB PROJECT

Project Name:

ServletTutorial

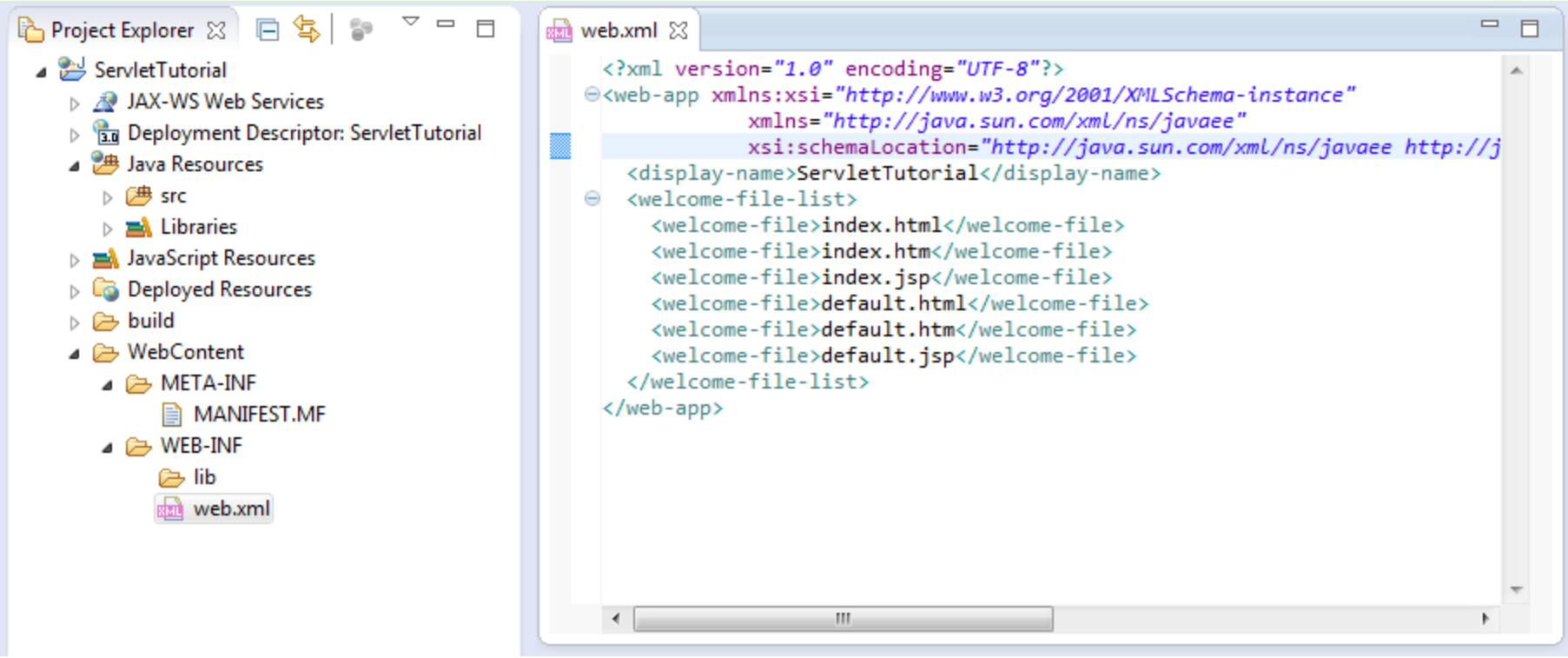


CREATING WEB PROJECT



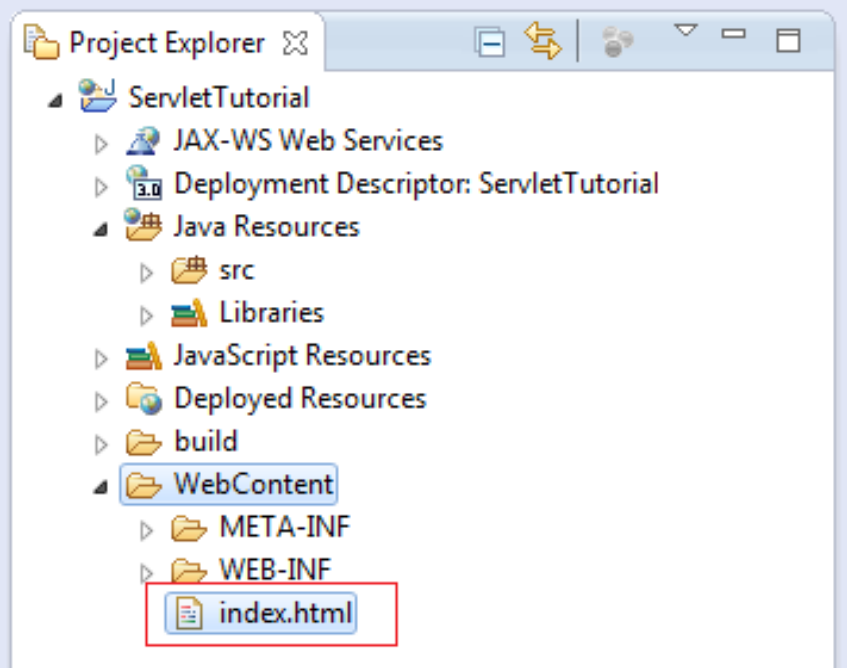
CREATING WEB PROJECT

- Project was created:



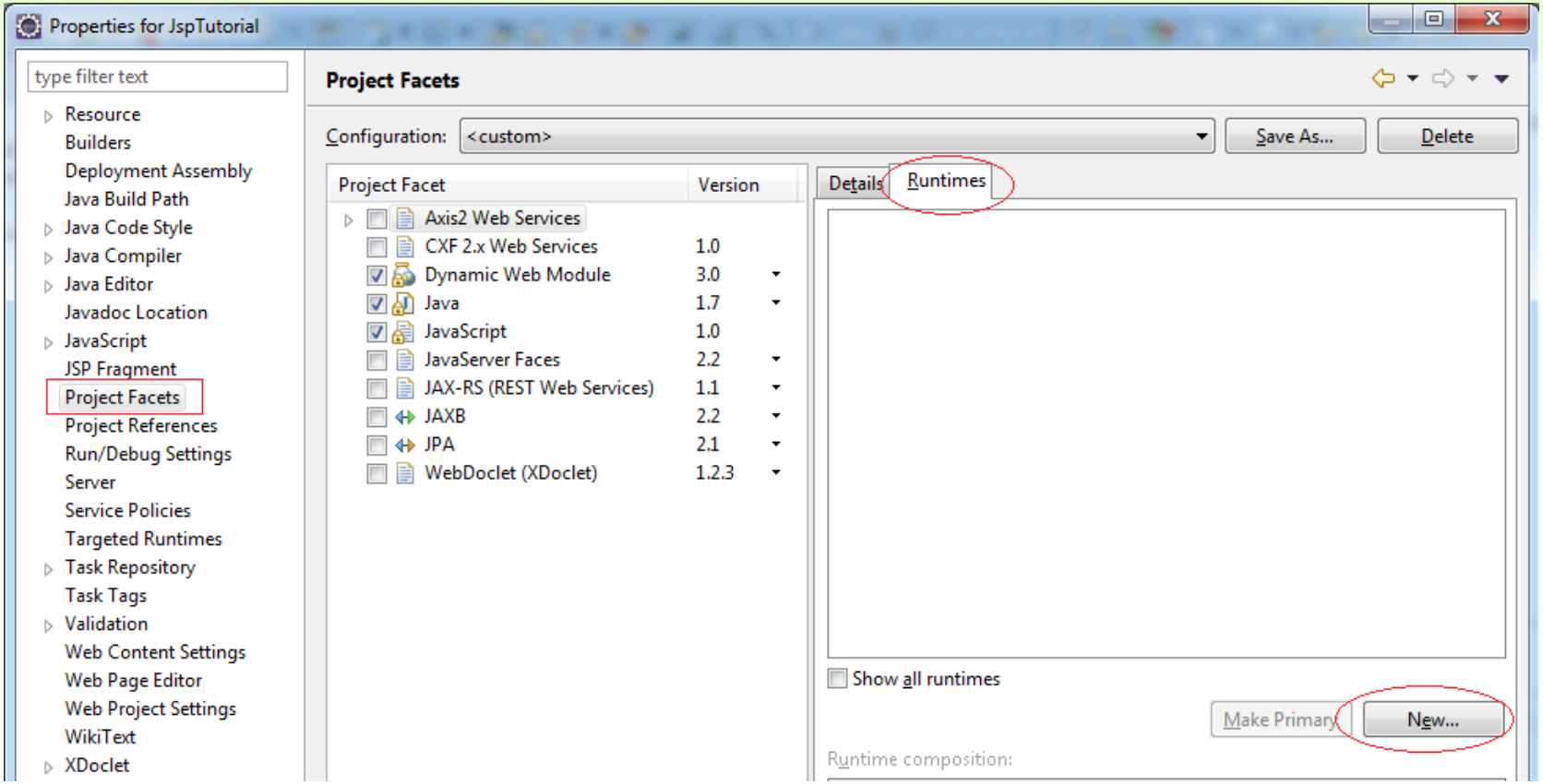
CREATING INDEX.HTML

❖Create file **index.html**:

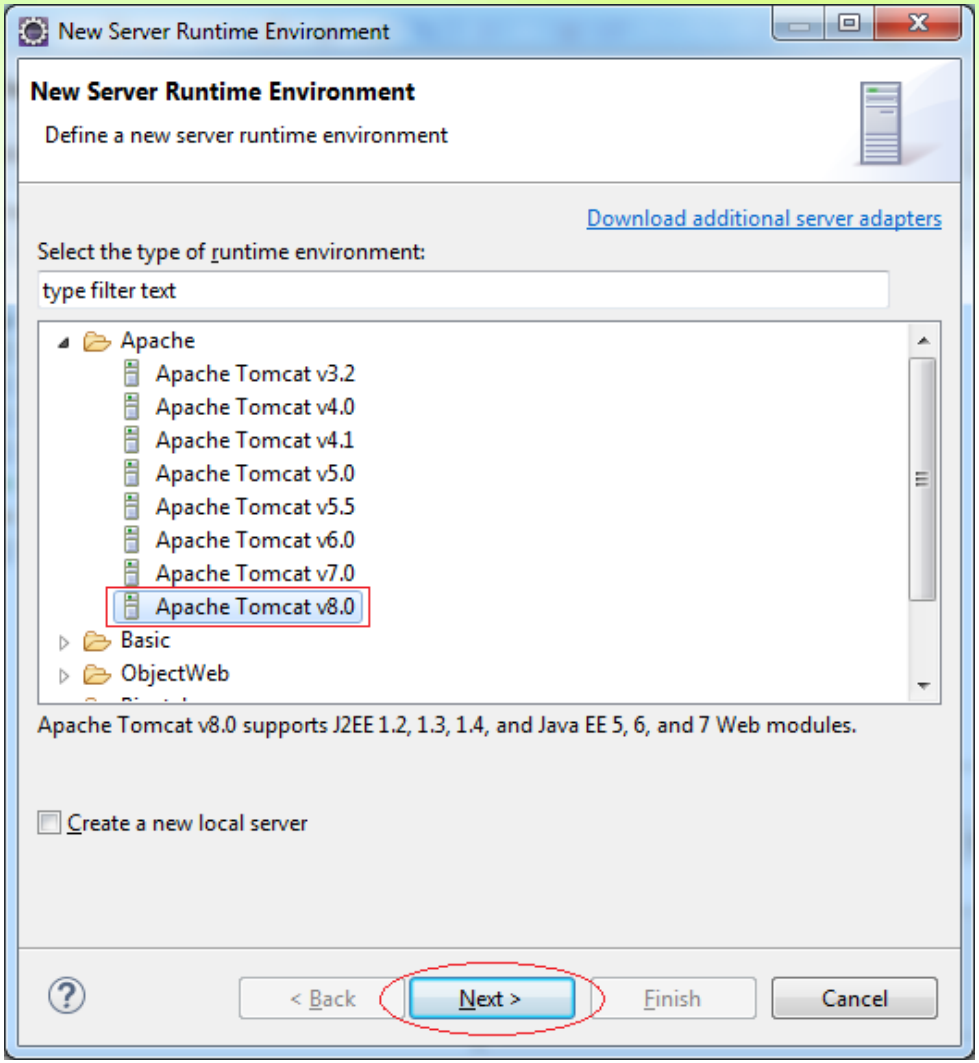


Configure Eclipse to run apps on Tomcat

In eclipse, right-click on Project Servlet Tutorial, select Properties

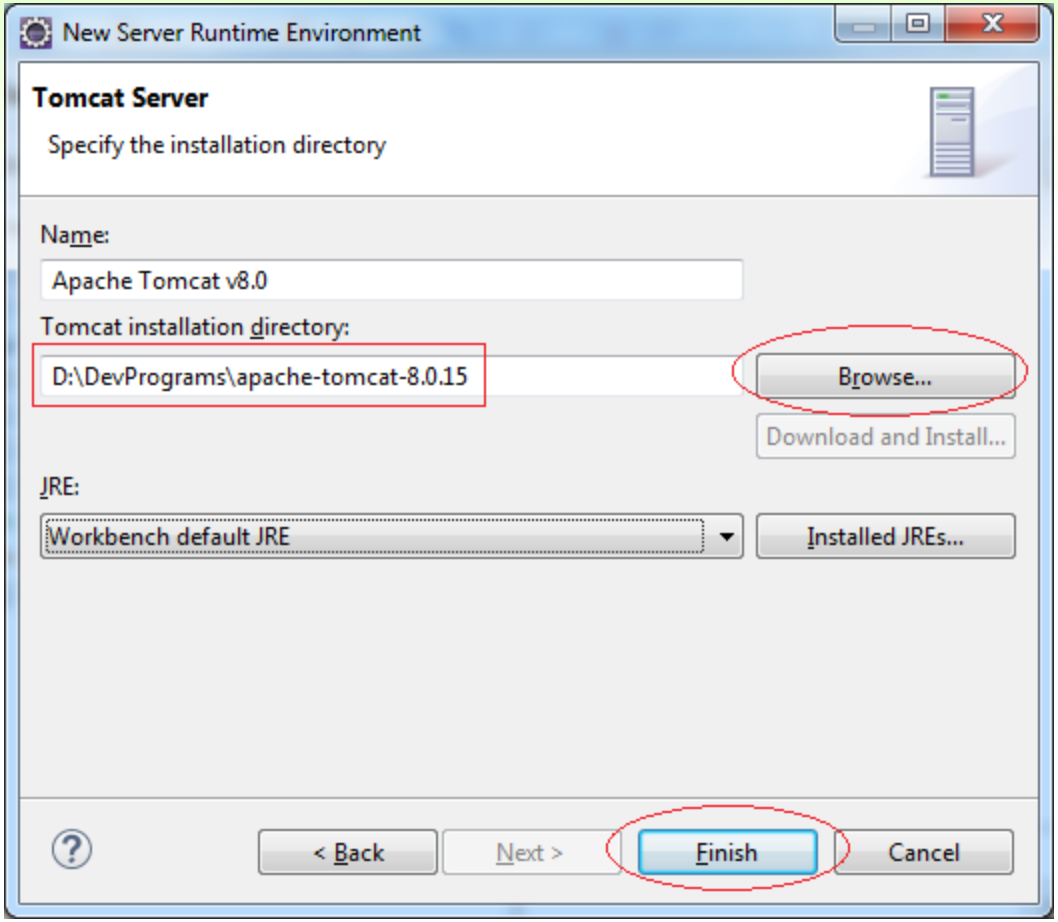


CONFIGURE ECLIPSE TO RUN APPS ON TOMCAT

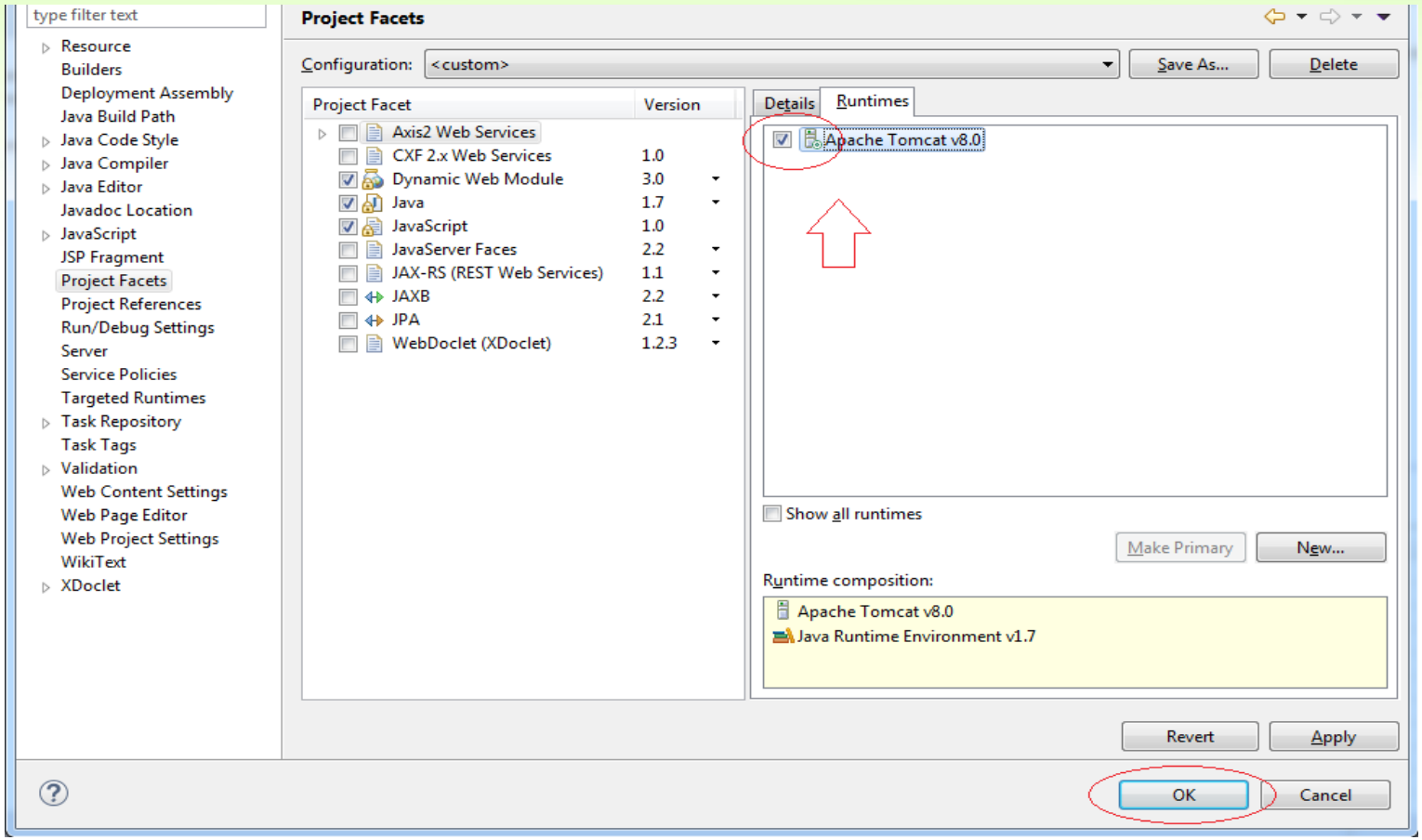


CONFIGURE ECLIPSE TO RUN APPS ON TOMCAT

Select the location where Tomcat is installed as shown in figure

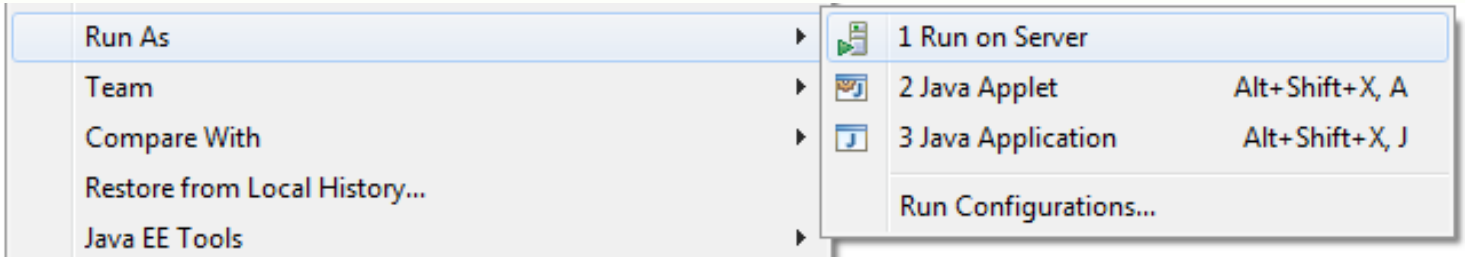


CONFIGURE ECLIPSE TO RUN APPS ON TOMCAT

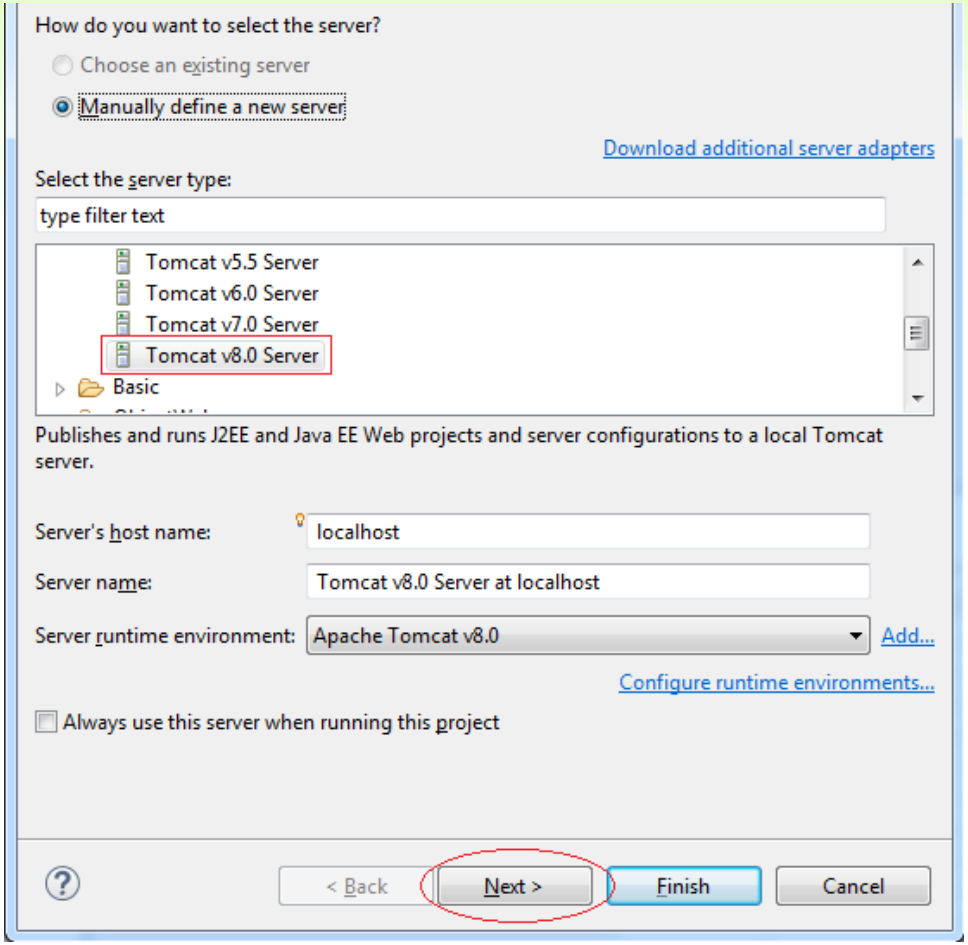


Configure Eclipse to run apps on Tomcat

Right-click the project ServletTutorial, select "Run As / Run on Server".



CONFIGURE ECLIPSE TO RUN APPS ON TOMCAT



CONFIGURE ECLIPSE TO RUN APPS ON TOMCAT

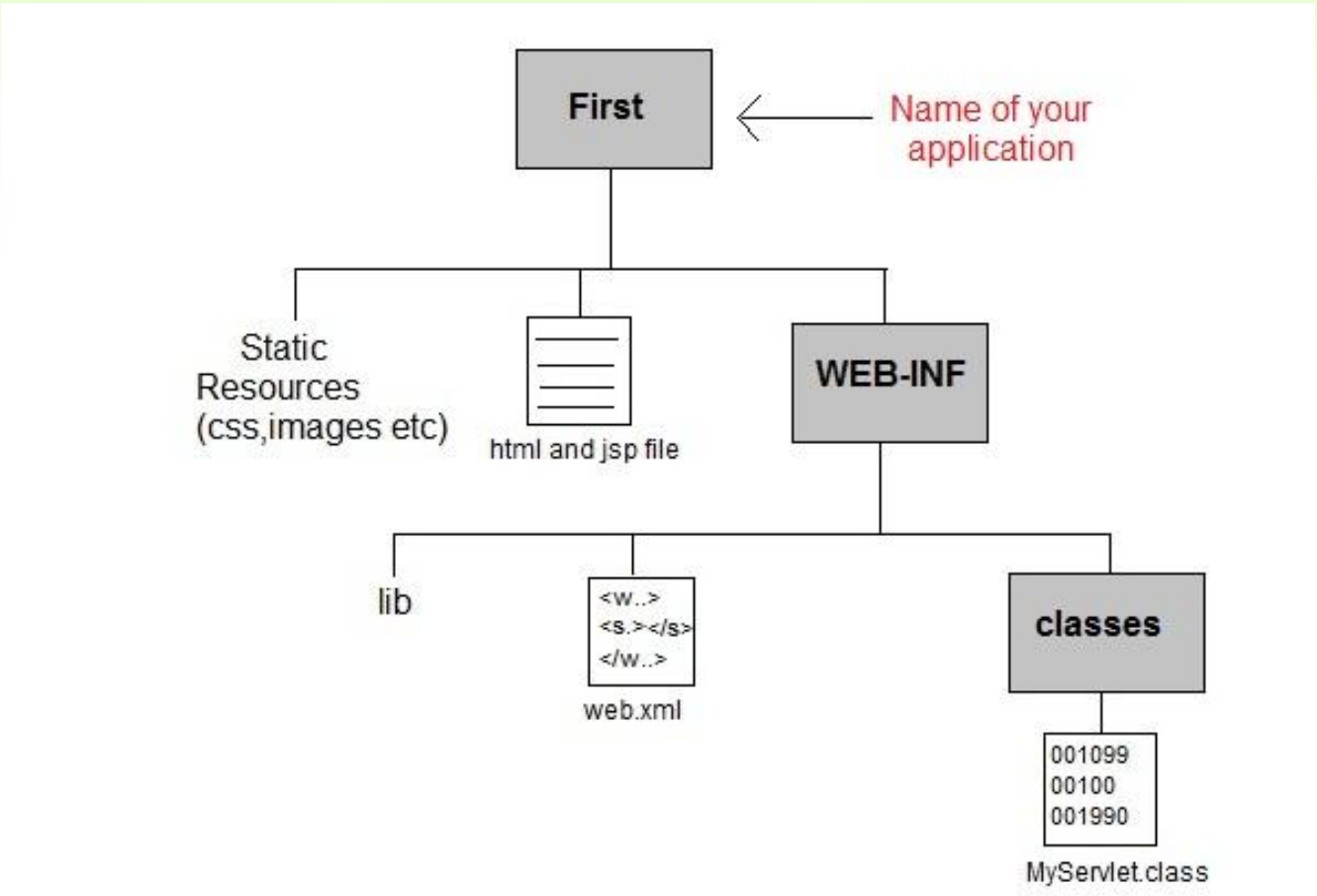
- Website is running on Eclipse browser.

Application With Tomcat Server

After installing Tomcat Server on your machine follow the below mentioned steps :

1. Create directory structure for your application.
2. Create a Servlet
3. Compile the Servlet
4. Create Deployment Descriptor for your application
5. Start the server and deploy the application

CREATING THE DIRECTORY STRUCTURE



Creating the Directory Structure

- All HTML, static files(images, css etc) are kept directly under Web application folder. While all the Servlet classes are kept inside classes folder.

- The **web.xml** (deployment descriptor) file is kept under WEB-INF folder.

Creating a Servlet

- There are three different ways to create a servlet.
  - By implementing Servlet interface
  - By extending GenericServlet class
  - By extending HttpServlet class

- But mostly a servlet is created by extending HttpServlet abstract class. HttpServlet gives the definition of service() method of the Servlet interface.

- The servlet class that we will create should not override service() method.

- Our servlet class will override only doGet() or doPost() method.

CREATING A SERVLET

❖When a request comes in for the servlet, the Web Container calls the servlet's service() method and depending on the type of request the service() method calls either the doGet() or doPost() method.

❖By default a request is Get request.

COMPILING A SERVLET

❖To compile a Servlet a JAR file is required. Different servers require different JAR files.

❖In Apache Tomcat server servlet-api.jar file is required to compile a servlet class.

❖Download servlet-api.jar file.

❖Paste the servlet-api.jar file inside Java\jdk\jre\lib\ext directory.

❖Compile the Servlet class.

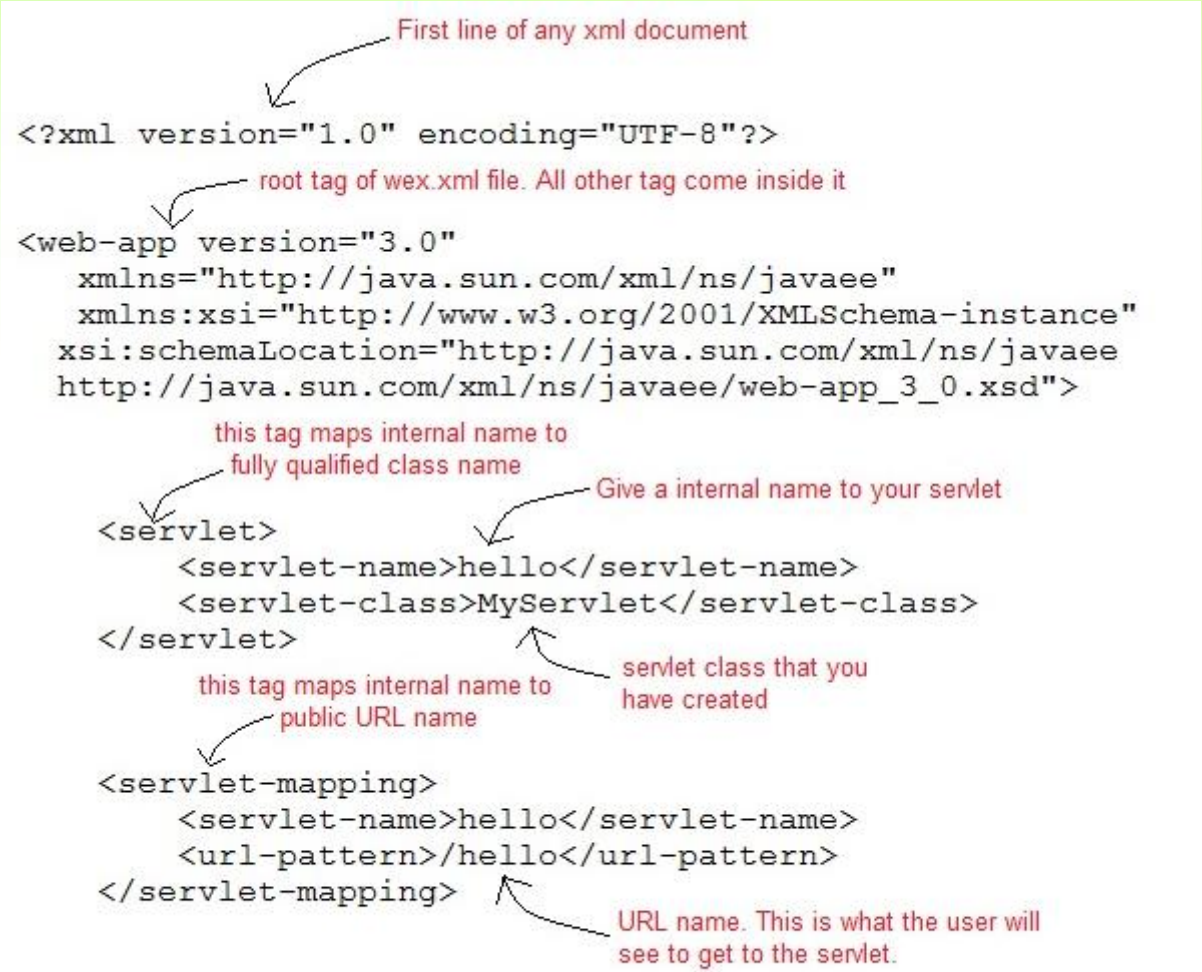
❖After compiling your Servlet class you will have to paste the class file into WEB-INF/classes/ directory.

CREATE DEPLOYMENT DESCRIPTOR

❖Deployment Descriptor(DD) is an XML document that is used by Web Container to run Servlets and JSP pages. DD is used for several important purposes such as:

- ❖Mapping URL to Servlet class.
- ❖Initializing parameters.
- ❖Defining Error page.
- ❖Security roles.
- ❖Declaring tag libraries.

CREATE DEPLOYMENT DESCRIPTOR



SERVLET REQUEST

❖True job of a Servlet is to handle client request. Servlet API provides two important interfaces **javax.servlet.ServletRequest** and **javax.servlet.http.HttpServletRequest** to encapsulate client request.

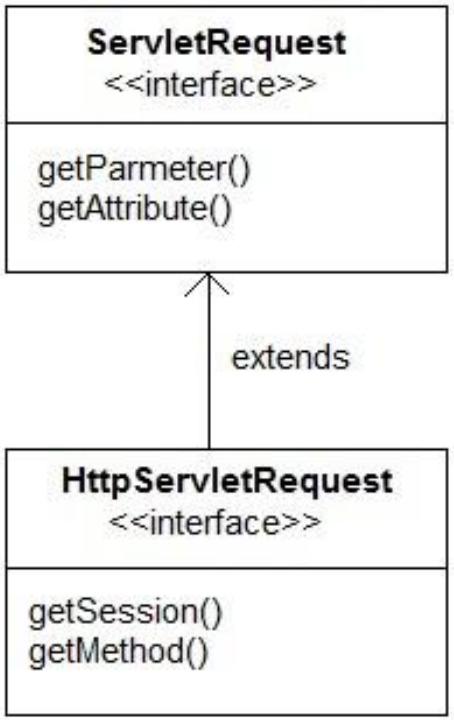
❖ Implementation of these interfaces provide important information about client request to a servlet.

METHODS OF SERVLET REQUEST

Methods	Description
Object `getAttribute(String name)`	return attribute set on request object by name
Enumeration `getAttributeName()`	return an Enumeration containing the names of the attributes available inthis request
int `getContentLength()`	return size of request body
int `getContentType()`	return media type of request content
ServletInputStream `getInputStream()`	returns a input stream for reading binary data
String `getParameter(String name)`	returns value of parameter by name
String `getLocalAddr()`	returns the Internet Protocol(IP) address of the interface on which the request was received
Enumeration `getParameterNames()`	returns an enumeration of all parameter names
String[] `getParameterValues(String name)`	returns an array of String objects containing all of the values the given request parameter has, or null if the parameter does not exist
ServletContext `getServletContext()`	return the servlet context of current request.
String `getServerName()`	returns the host name of the server to which the request was sent

HTTP SERVLET REQUEST INTERFACE

❖**HttpServletRequest** interface adds the methods that relates to the HTTP protocol.



METHODS OF HTTP SERVLET REQUEST

Methods	Description
String `getContextPath()`	returns the portion of the request URI that indicates the context of the request
Cookies `getCookies()`	returns an array containing all of the Cookie objects the client sent with this request
String `getQueryString()`	returns the query string that is contained in the request URL after the path
HttpSession `getSession()`	returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session
String `getMethod()`	Returns the name of the HTTP method with which this request was made, for example, GET, POST, or PUT.
Part `getPart(String name)`	gets the Part with the given name
String `getPathInfo()`	returns any extra path information associated with the URL the client sent when it made this request.

SERVLET RESPONSE

❖Servlet API provides two important interfaces **ServletResponse** and **HttpServletResponse** to assist in sending response to client.

Methods	Description
PrintWriter `getWriter()`	returns a PrintWriter object that can send character text to the client.
void `setBufferSize(int size)`	Sets the preferred buffer size for the body of the response
void `setContentLength(int len)`	Sets the length of the content body in the response In HTTP servlets, this method sets the HTTP Content-Length header
void `setContentType(String type)`	sets the content type of the response being sent to the client before sending the respond.
void `setBufferSize(int size)`	sets the preferred buffer size for the body of the response.
boolean `isCommitted()`	returns a boolean indicating if the response has been committed
void `setLocale(Locale loc)`	sets the locale of the response, if the response has not been committed yet.

HTTP SERVLET RESPONSE INTERFACE

❖HttpServletResponse interface adds the methods that relates to the HTTP response.

Methods	Description
void `addCookie(Cookie cookie)`	adds the specified cookie to the response.
void `sendRedirect(String location)`	Sends a temporary redirect response to the client using the specified redirect location URL and clears the buffer
int `getStatus()`	gets the current status code of this response
String `getHeader(String name)`	gets the value of the response header with the given name.
void `setHeader(String name, String value)`	sets a response header with the given name and value
void `setStatus(int sc)`	sets the status code for this response
void `sendError(int sc, String msg)`	sends an error response to the client using the specified status and clears the buffer

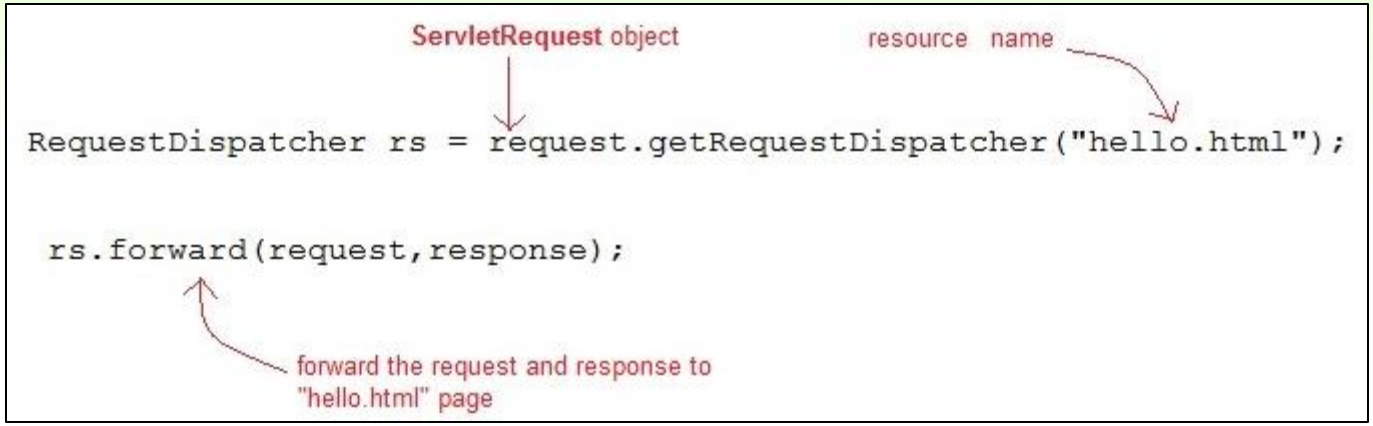
REQUEST DISPATCHER

❖RequestDispatcher is an interface, implementation of which defines an object which can dispatch request to any resources(such as HTML, Image, JSP, Servlet) on the server.

Methods	Description
void `forward(ServletRequest request, ServletResponse response)`	forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server
void `include(ServletRequest request, ServletResponse response)`	includes the content of a resource (servlet, JSP page, HTML file) in the response

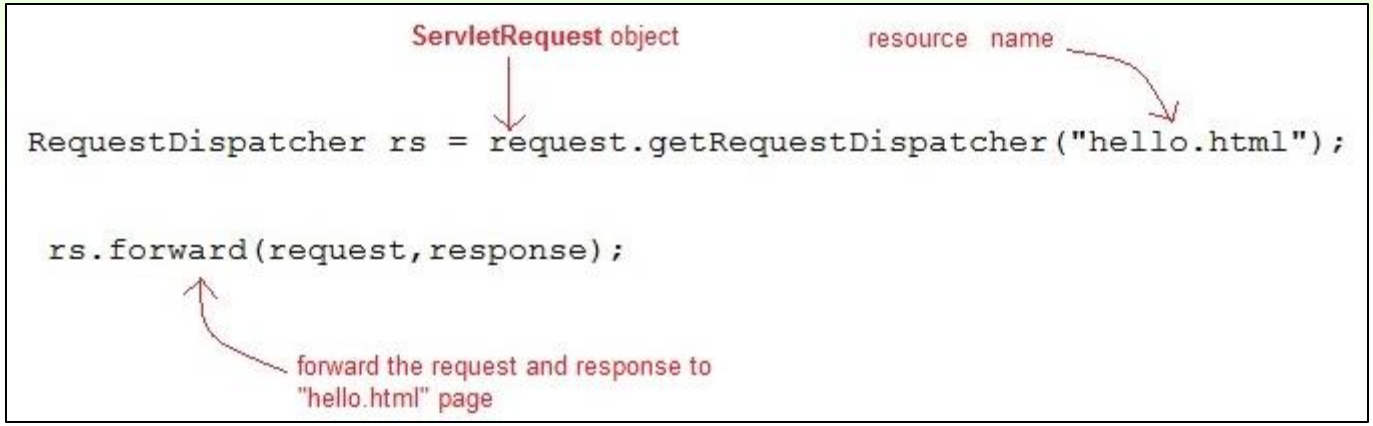
OBJECT OF REQUEST DISPATCHER

❖**getRequestDispatcher()** method of ServletRequest returns the object of RequestDispatcher.



OBJECT OF REQUEST DISPATCHER

❖**getRequestDispatcher()** method of ServletRequest returns the object of RequestDispatcher.



LAB 1

- Create a simple **login page** with username and password

- On clicking submit, check if the user has entered correct password or not

- If the password is correct, then redirect the user to a new page and show the user name

THANK YOU

http://www.4kitsolutions.com

By
Sudha Agarwal
sudha.agarwal@4kitsolutions.com