



Let's Solve



Celebrating 20 Years

Servlet & JSP



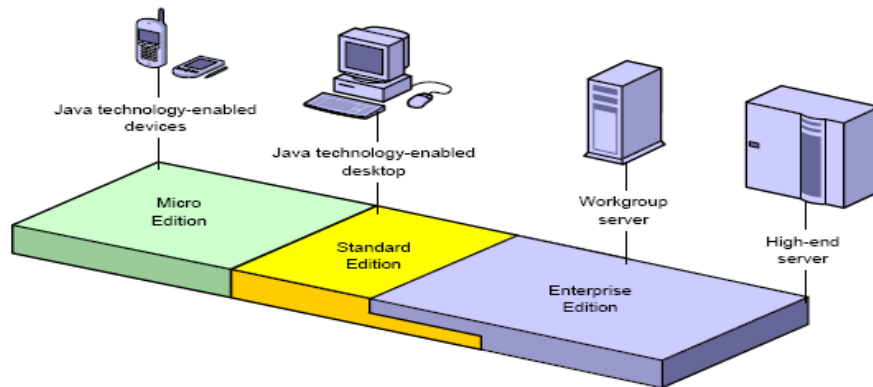
Larsen & Toubro
Group Company

The Java Enterprise Edition

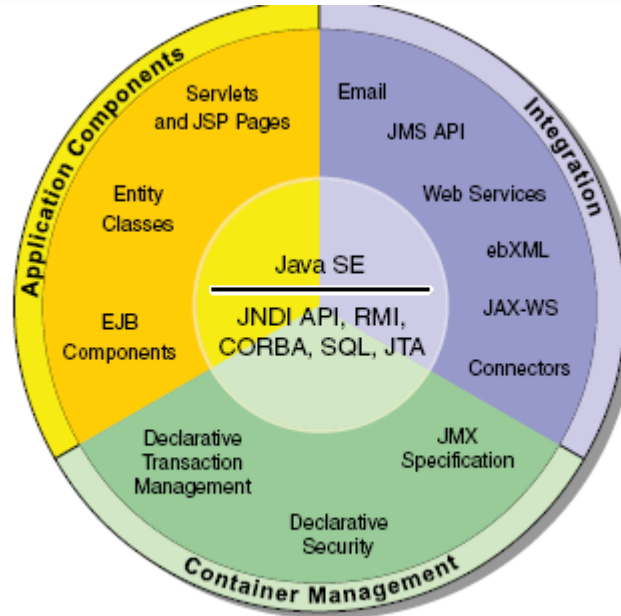
- The Java EE platform:
 - Is an architecture for implementing enterprise-class applications.
 - Uses Java and Internet technology
 - Has a primary goal of simplifying the development of enterprise-class applications through an application model that is:
 - Vendor-neutral
 - Component-based

Introduction to Java EE

- Java Enterprise Edition addresses concerns related to the usage of Java on the server side. It's a standard specification so that vendors interested in providing support for Java on their servers confirm to it
- Servlets, JSP, EJB, WebServices, Transactions, Messaging, ... are all part of this specification
- Software providing support for JEE are referred to as Application Servers
 - WebSphere, WebLogic, JBoss, Glassfish, ...



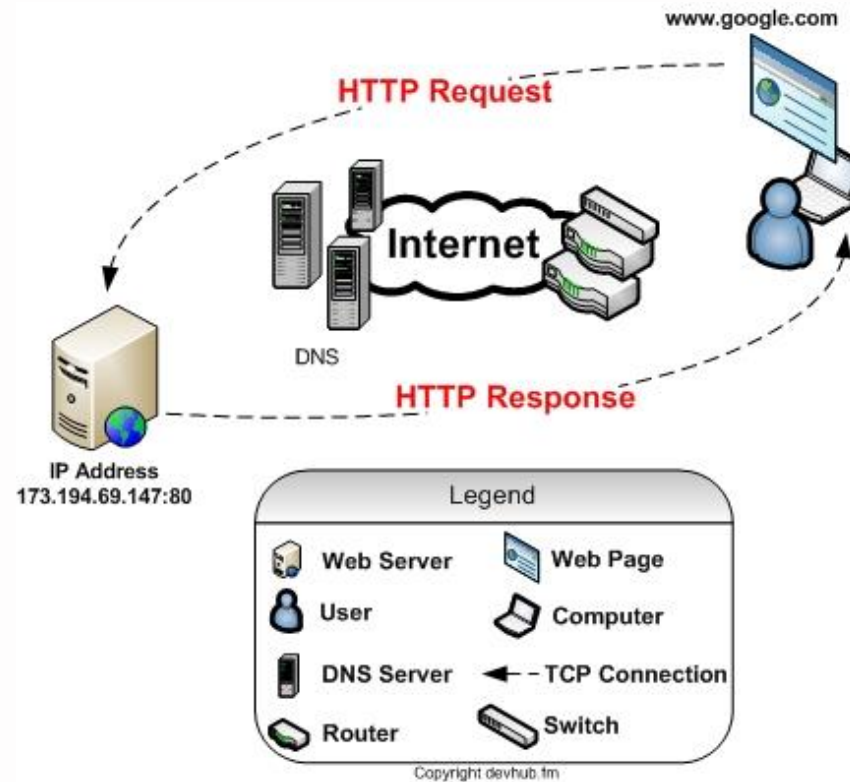
Java EE Technology Suite



Developing Web applications

- Almost every application developed today is web based, i.e browser based. All the user needs to do is type some URL and the application loads within the browser in hardly any time
- Web Servers provide all the support we need to provide service to multiple users accessing the website. The protocol used between the browser and webserver is HTTP by default
- HTTP protocol is stateless, which means everytime we send a request, there is no trace of the previous request on the server
- HTTP protocol is based on the paradigm of request-response style communication. Client sends a request to the server and the server replies back with the response accordingly

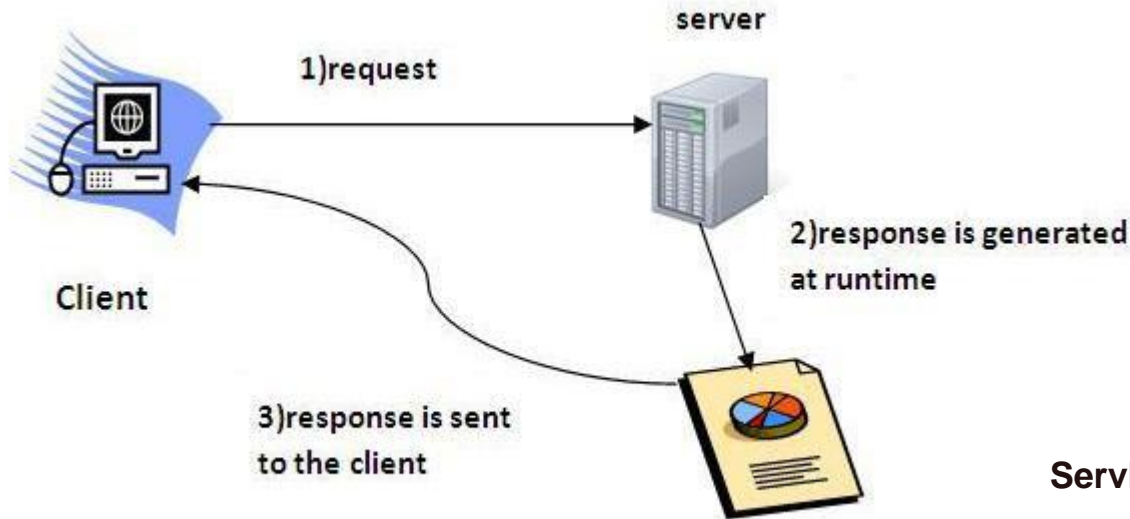
HTTP Request Response model



Developing Web applications

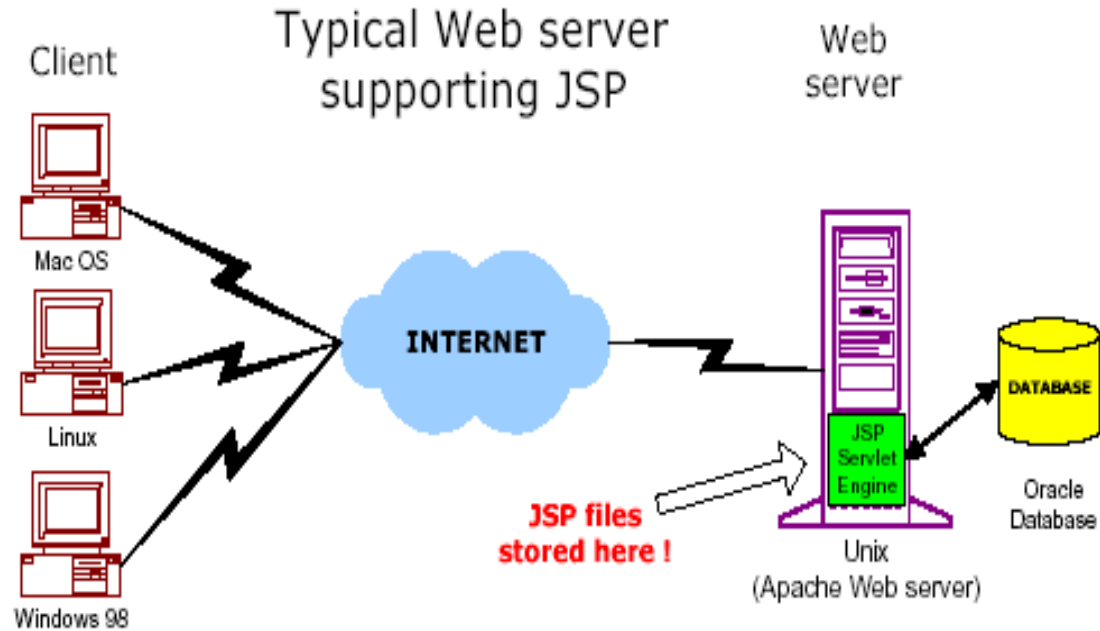
- The most important feature of a Web Server is, it allows us to extend it's capability by writing server side extensions
- These extensions are written using the API provided by the web server
- Servlets and JSP give us the same power of Java on the server side whereby allowing us to write code independent of any web server
- This is achieved using a separate software called as Web Container/Servlet Container
 - Tomcat/JRUN, ...

Servlet/JSP



Servlet/JSP

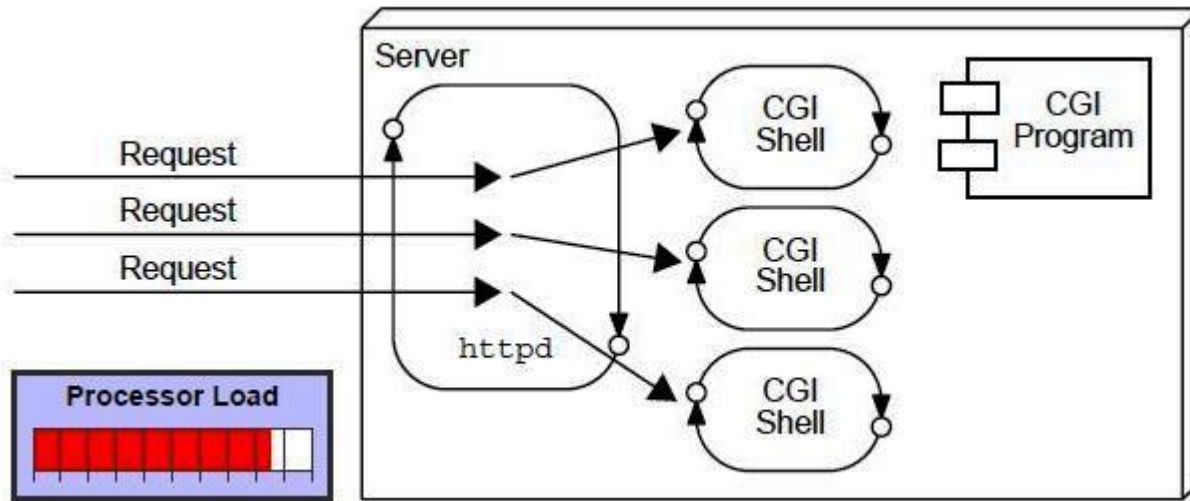
Web Server and Servlet/JSP



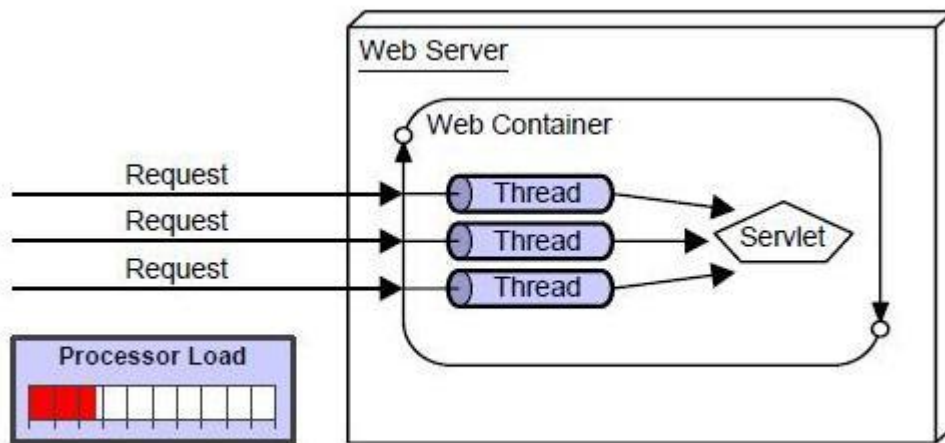
Servlet/JSP features

- In simple words, Servlet/JSPs are Java programs which run on the server side whenever any request for the same is submitted by the client
- All server side frameworks share a common set of features:
 - Read data submitted by the user
 - Generate HTML dynamically based on user input
 - Determine information about the client browser
 - Access Database systems
 - Exploit the HTTP protocol

CGI Architecture

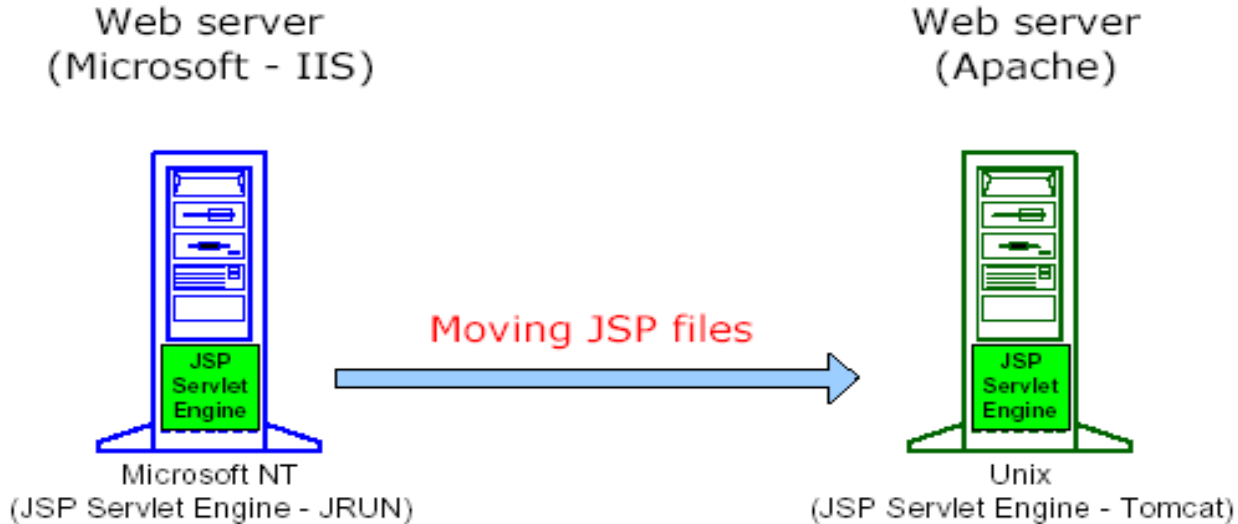


Servlet/JSP Architecture



Servlets and JSP are portable

Moving JSP file from one platform to another.



Why build pages dynamically?

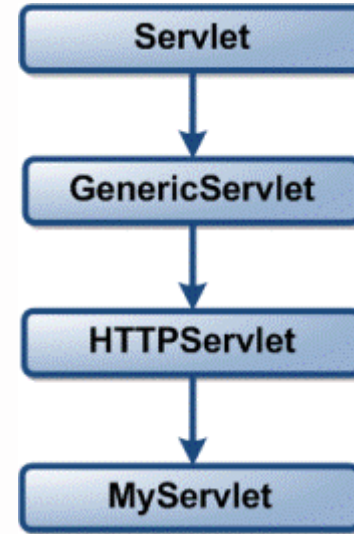
- The Web page is based on data submitted by the user
 - E.g., results page from search engines and order-confirmation pages at online stores
- The Web page is derived from data that changes frequently
 - E.g., a weather report or news headline page
- The Web page uses information from databases or other server side resources
 - E.g., an e-commerce site could use a servlet to build a Web page that lists the current price and availability of each item that is for sale

Technologies used for developing Web Applications

- HTML is the primary technology used over the Web
- To add dynamism in HTML pages, we use Javascript, DOM and DHTML
- To manage the layout and appearance of Web pages, we use CSS
- To dynamically generate HTML pages, we use Server side technologies like Servlets, JSP, PHP, ASP.NET and others
- To asynchronously communicate with the server, we use AJAX

The Servlet API

- The *javax.servlet* package provides interfaces and classes for writing servlets
- All servlets implement Servlet interface either directly or more commonly, by extending a class that implements it such as HttpServlet
- The Servlet interface declares, methods that manage the servlet and its communications with clients
- Servlet writers provide some or all of these methods when developing a servlet



The Servlet API

- javax.servlet package contains:
 - ServletRequest Interface
 - ServletResponse Interface
 - ServletContext Interface
- javax.servlet.http package contains:
 - HttpServletRequest Interface
 - HttpServletResponse Interface
 - HttpSession Interface

Servlet Life Cycle

- init
 - Executed once when the servlet is first loaded. Not called for each request
- service
 - Called in a new thread by server for each request. Dispatches to doGet, doPost, etc...
- doGet, doPost, etc...
 - Handles HTTP GET, POST, etc... requests
- destroy
 - Called when server deletes the servlet instance. Not called after each request

Deployment Descriptor

- /WEB-INF/web.xml
 - Part of the standard
 - Defines servlets used in the web application
 - Maps servlets to URLs
 - A servlet can map to many URLs
- Defines resources available to the web app
- Defines security constraints
- Defines other stuff like
 - Welcome file list
 - Session timeout
 - Error page mapping

web.xml example

```
<?xml version="1.0" encoding="UTF-8"?> <!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/web-app_2_3.dtd">
```

```
<web-app>
```

```
<servlet>
```

```
  <servlet-name>FirstServlet</servlet-name>
```

```
  <servlet-class>com.training.FirstServlet</servlet-class>
```

```
</servlet>
```

```
<servlet-mapping>
```

```
  <servlet-name>FirstServlet</servlet-name>
```

```
  <url-pattern>/first</url-pattern>
```

```
</servlet-mapping>
```

```
</web-app>
```

Example

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

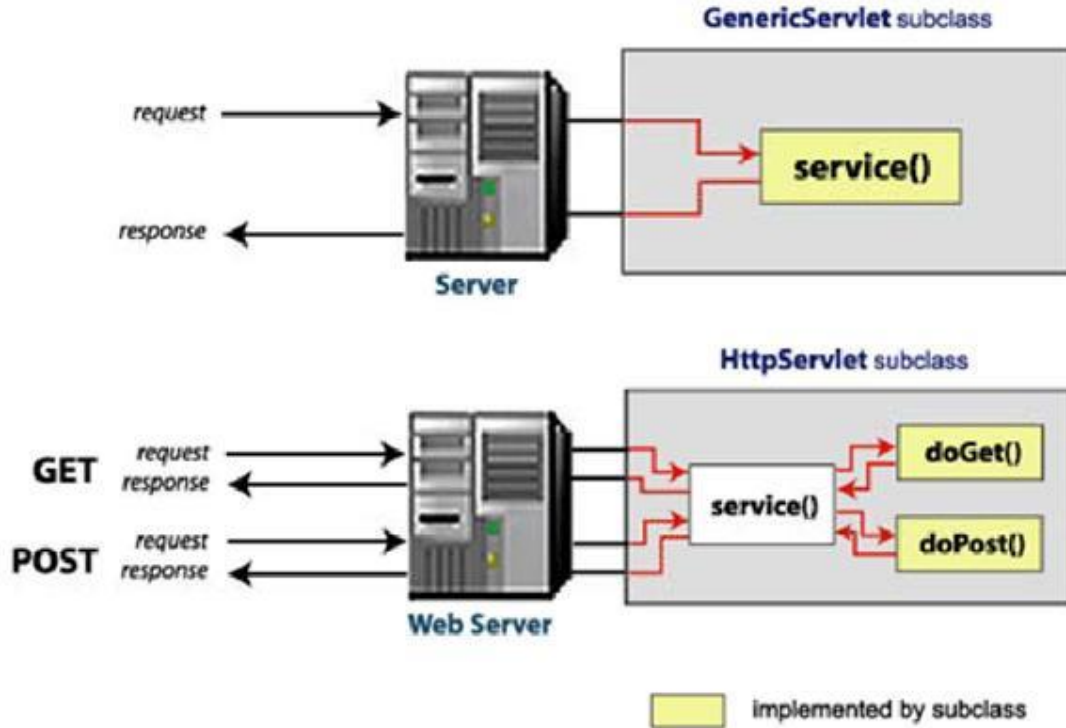
public class FirstServlet extends HttpServlet {

    public void doGet ( HttpServletRequest request,
                        HttpServletResponse response )
                        throws ServletException, IOException {

        PrintWriter out =response.getWriter();
        response.setContentType("text/html");
        out.println("<HTML><BODY>");
        out.println("This is output from SimpleServlet.");
        out.println("</BODY></HTML>");

    }
}
```

Cont'd..

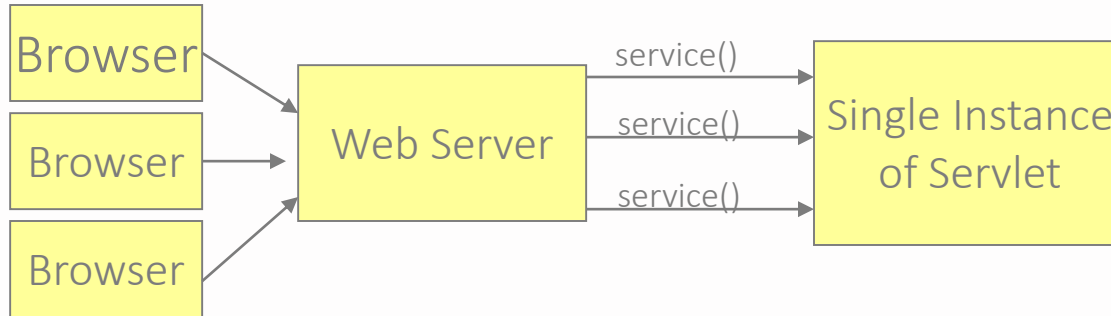


The init() method

- The init() method is called when the servlet is first requested by a browser request
- It is not called again for each request
- Used for one-time initialization
- The init() method is a good place to put any initialization variables

The service() method

- Each time the server receives a request for a servlet, the server spawns a new thread and calls the servlet's service () method



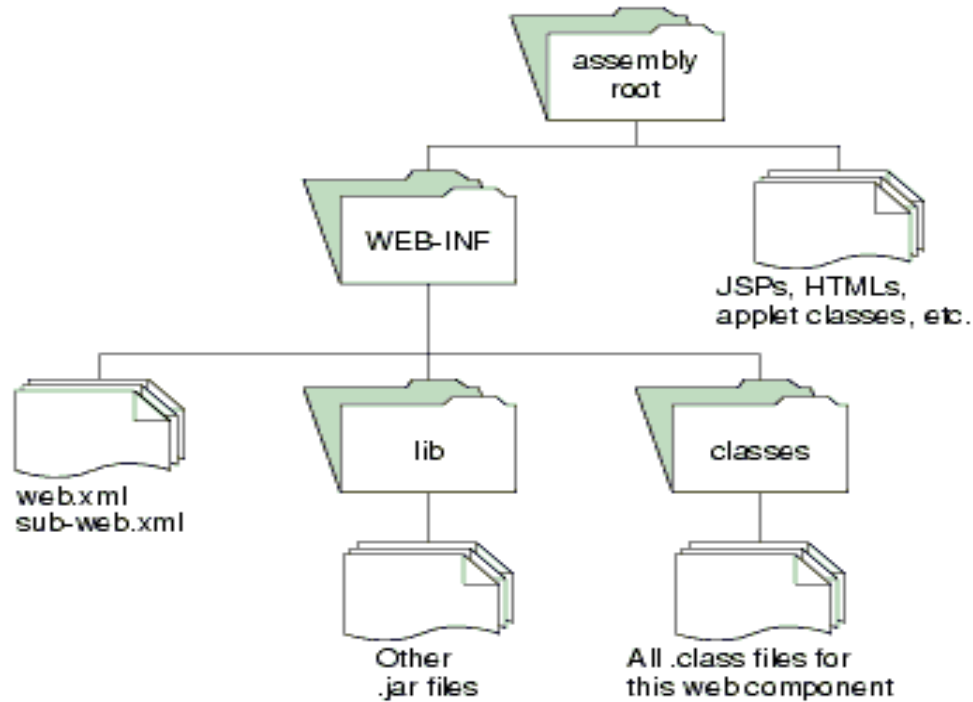
The service() method

- By default the service() method checks the HTTP Header
- Based on the header, service calls either doPost() or doGet()
- doPost and doGet is where you put the majority of your code
- If your servlets needs to handle both get and post identically, have your doPost() method call doGet() or vice versa

Servlet destroy() method

- Before a server shuts down, it will call the servlet's destroy() method
- You can handle any servlet clean up here. For example:
 - Updating log files
 - Closing database connections
 - Closing any socket connections

Structure of a Web Module



Scopes in Servlet/JSP

- Enables sharing information among collaborating web components
- The four different scopes are:
 - Context (ServletContext) scope
 - Also called as application scope
 - User for sharing global data across all the components in the application
 - Session scope
 - Used for storing user specific information on the server
 - Request scope
 - Used for passing information from one servlet/jsp to another servlet/jsp when using forwarding/servlet chaining
 - Page scope
 - Used exclusively in JSPs for storing information visible only to the content within the page

ServletContext interface

- Defines a set of methods that a servlet uses to communicate with its servlet container
- There is one context per "web application" and is accessible to all active resource of that application
- The ServletContext object enables to set, get and change Web application-scope attribute values
- Used to get Server Information , like name, version of the container and the API being used

Initialization parameters in Servlet/JSP

- There are two `InitParameter()` methods
 - The Servlet init parameter
 - The Context init parameter
- **Servlet Init Parameter:**
 - When the container initializes a servlet it makes a unique `ServletConfig` object for the servlet
 - The container reads the servlet init parameters from the DD and gives them to the `ServletConfig` to the Servlet's `init()` method
 - `String emailid=getServletConfig().getInitParameter("adminid");`**
 - Available to only the servlet for which the `<init-param>` was configured

```
<servlet>
<init-param>
  <param-name>adminid</param-name>
  <param-value>sri@123.com</param-value>
</init-param>
</servlet>
```

Cont'd...

- Context Init Parmeter:

- Its defined within the <web-app> element but not within a specific servlet
- Available to any servlet and jsp that's part of this web applicaiton

```
String dsrsrc = getServletContext().getInitParameter("datasource");
```

```
<web-app>
```

```
  <context-param>
```

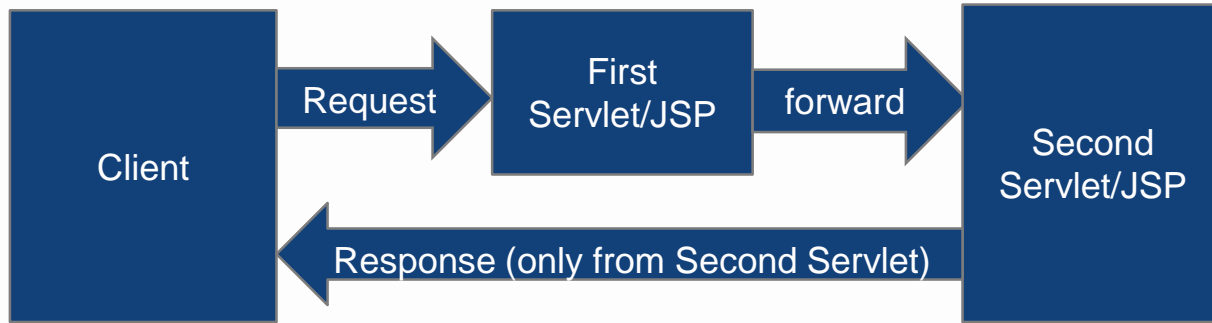
```
    <param-name>datasource</param-name>
```

```
    <param-value>myoracle</param-value>
```

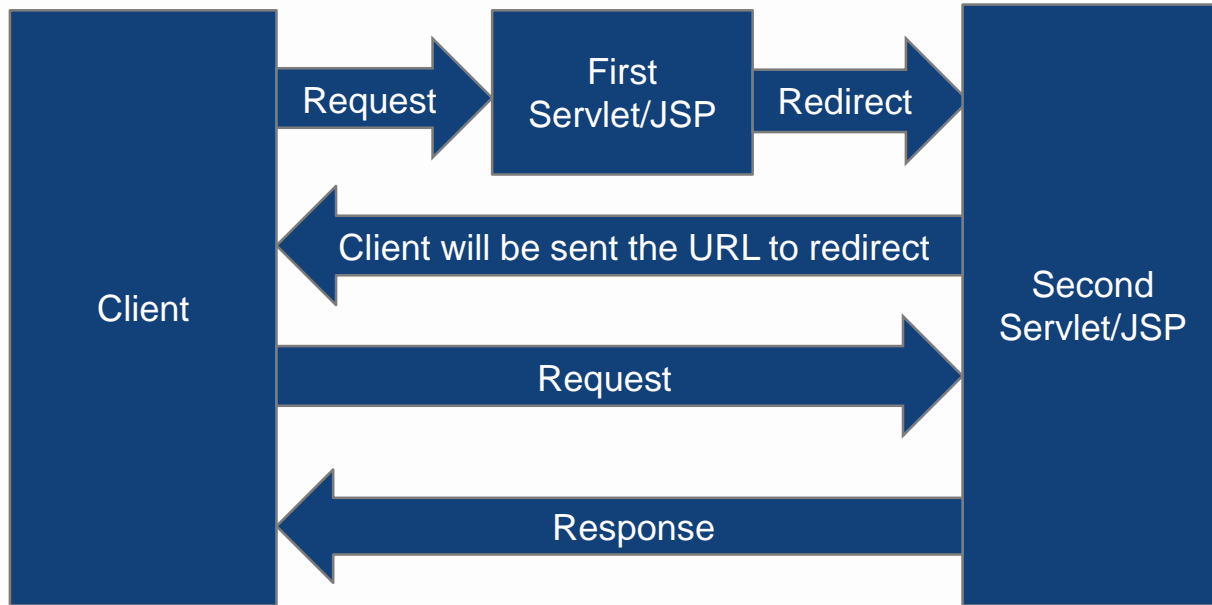
```
  </context-param>
```

```
</webapp>
```

How forwarding works



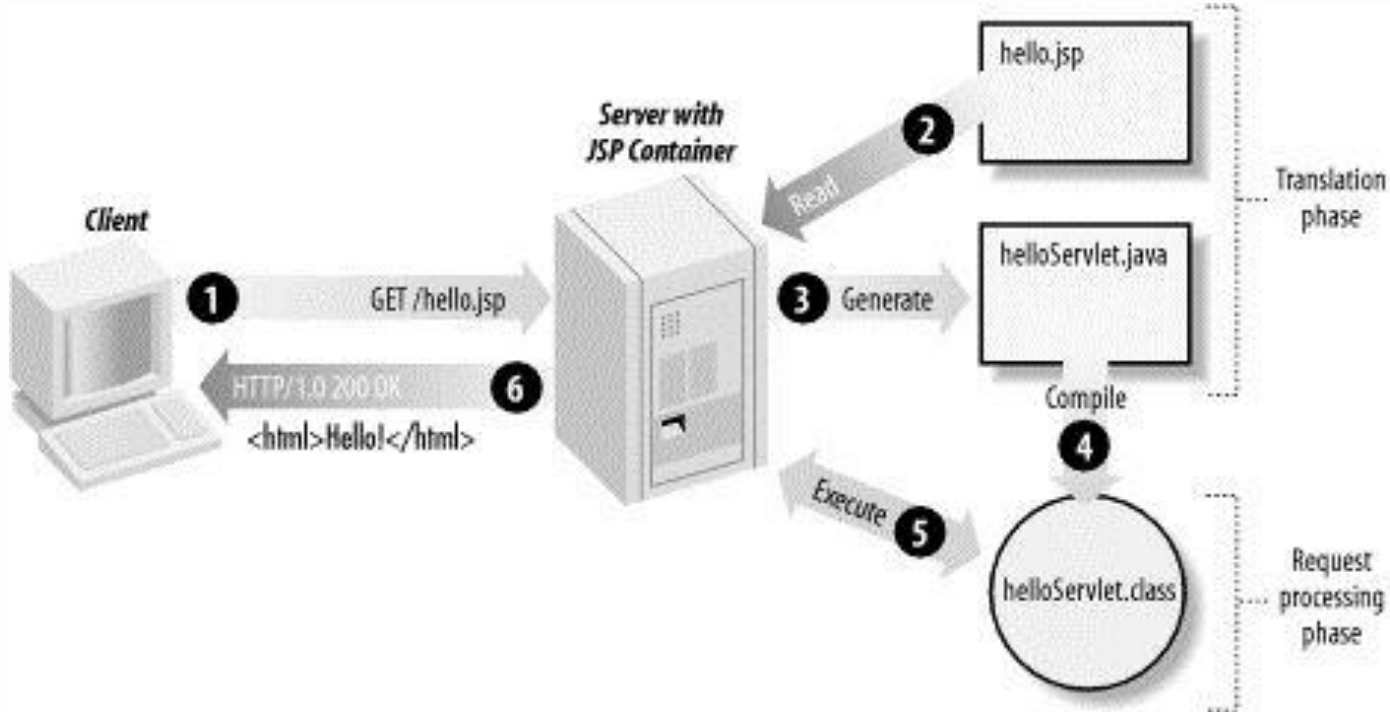
How Redirection works



Java Server Pages (JSP)

- Alternative to Servlets
- Servlets are cumbersome as it's a pure Java class with HTML embedded in it
- JSP is an HTML or an XML or any other markup file with Java code embedded in it
- When requested, JSP is compiled into a Servlet
- Tag libraries are a big plus
- JSTL, JSF, Struts, EL, etc... add more power

JSP Architecture

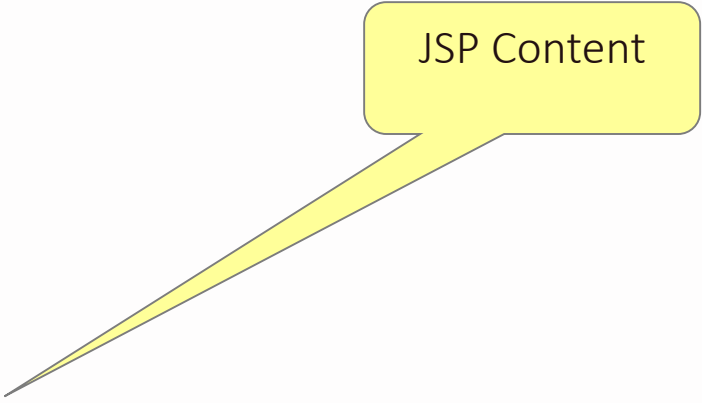


JSP lifecycle methods

- `jspInit()`: Is invoked at the time when the servlet is initialized
- `_jspService()`: Is invoked when request for the JSP page is received
- `jspDestroy()`: Is invoked before the servlet is removed from the service

Example JSP

```
<html>
<head>
<title>Java Server Page</title>
</head>
<body>
<center>
<%
out.println("<b>Hello From Simple JSP Page</b>");
%>
</center>
</body>
</html>
```



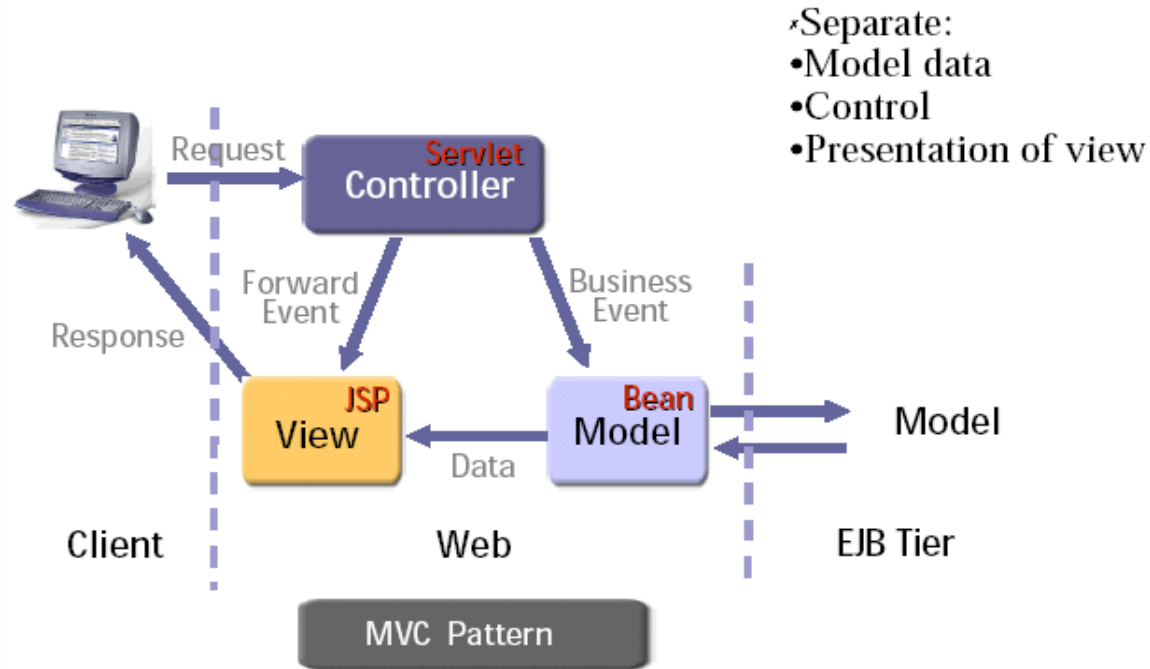
JSP Content

Session management

- Since HTTP is stateless in nature, explicit efforts need to be taken for managing conversational state between client and server
- Session management is the technique by which we keep information about the user accessing the site for various reasons:
 - For ex:
 - Maintaining logged in user's info
 - Maintaining shopping cart details
 - ...
- A Session will be created when a connection to the server is established and will be closed when the connection is closed. Each Session has a unique Session-Id associated with it

`HttpSession session = request.getSession();`

Servlet, JSP and MVC



- JSTL (JSP Standard Tag Libraries) is a collection of JSP custom tags. JSTL allows you to program your JSP pages using tags, rather than the scriptlet code that most JSP programmers are already accustomed to. JSTL can do nearly everything that regular JSP scriptlet code can do
- Example using Scriptlet:

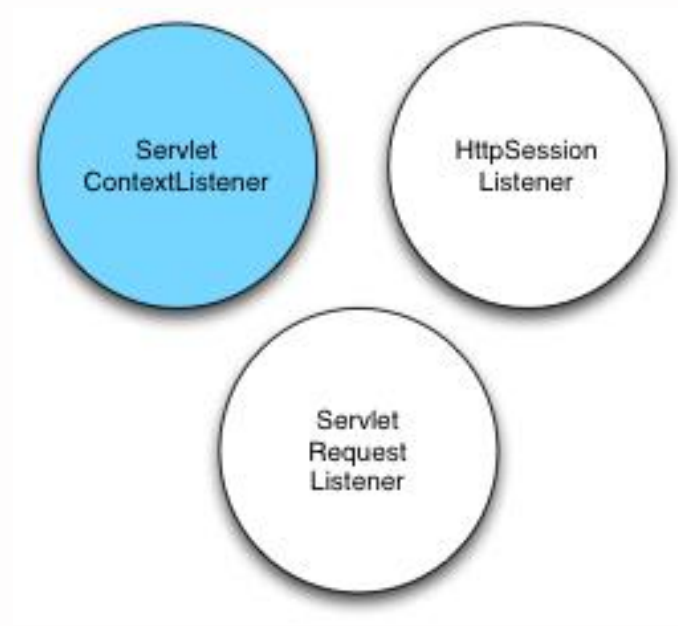
```
<body>
<% for(int i=1;i<=10;i++) { %>
    <%=i %><br/>
<%
    }
%>
</body>
```

Example using JSTL:

```
<body>
<c:forEach var="i" begin="1" end="10" step="1">
<c:out value="${i}" />
</c:forEach>
</body>
```


Servlet Listeners

- Monitoring and reacting to events in a servlet's life cycle can be done by defining listener objects
- The Methods get invoked when life-cycle events occur

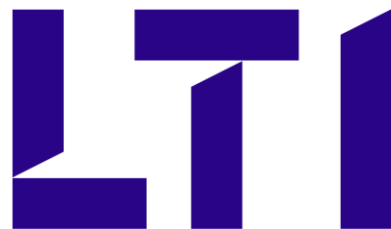


Filters

- Transform the content of HTTP requests, response, and header information
- Do not generally create response or responding to requests
- Attached to one or more servlets or JSP
- Modify or adapt the request/response for a resource
- Act on dynamic or static content – web resources

Configuring Authentication

- The Three types authentication that are recognized by the Java EE platform specification are
 - **HTTP Basic**
 - Specify this type in the web.xml deployment descriptor
 - **Client certificate**
 - Specify this type of authentication in web.xml Configure the web container's certificate database and assign trust levels per vendor instructions
 - **Form-based**
 - Specify this type in web.xml
 - Create and package the login page and error page
 - Specify names of these pages in web.xml



Let's Solve