Mindtree

Welcome to possible

Servlet

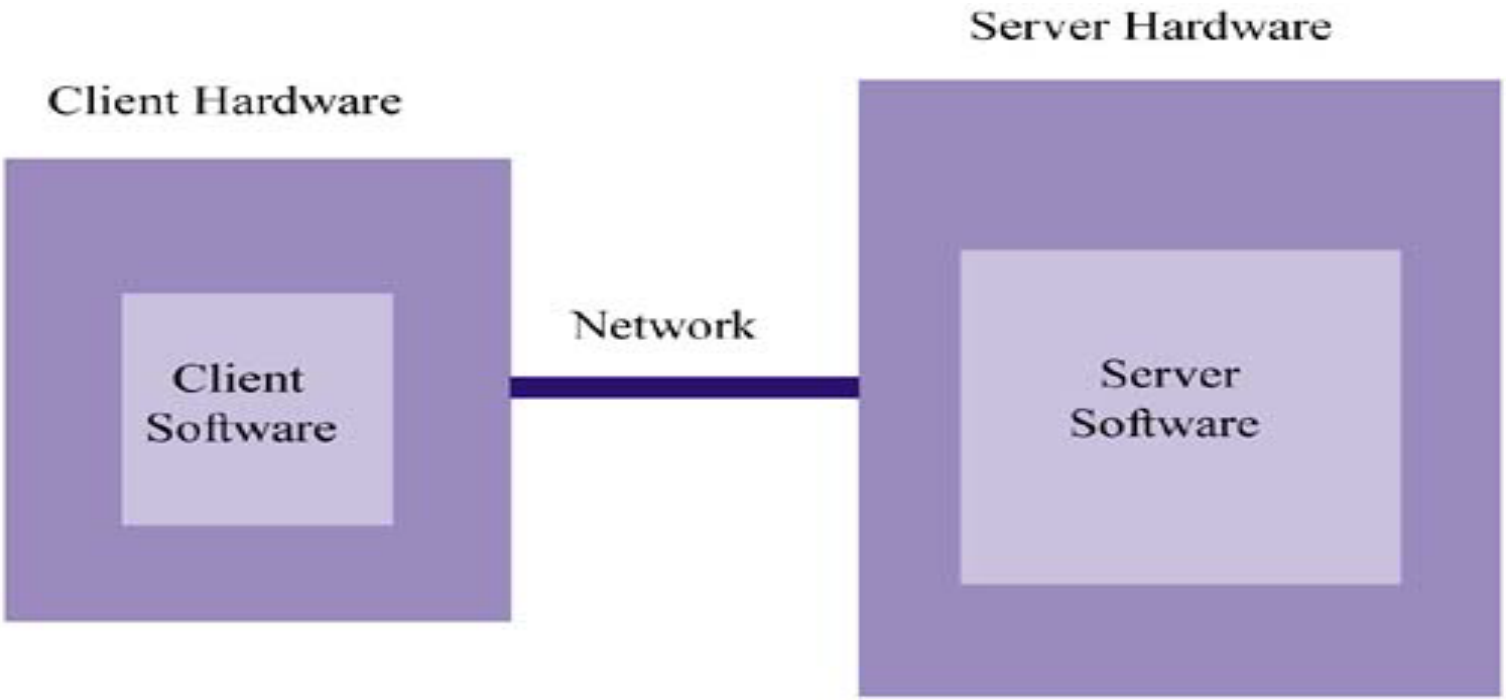
C2 - Orchard

Objectives

- Define Web application
- Understand Servlet Technology
- Understand how to write a simple servlet to generate dynamic content.
- Understand the Web application folder structure
- Understand the basic form of deployment descriptor
- Understand setting MIME type
- Understand PrintWriter and ServletOutputStream.
- Understand how to handle GET and POST requests
- Understand HttpServletRequest and HttpServletResponse
- Understand how to read form data in Servlet.

Web application

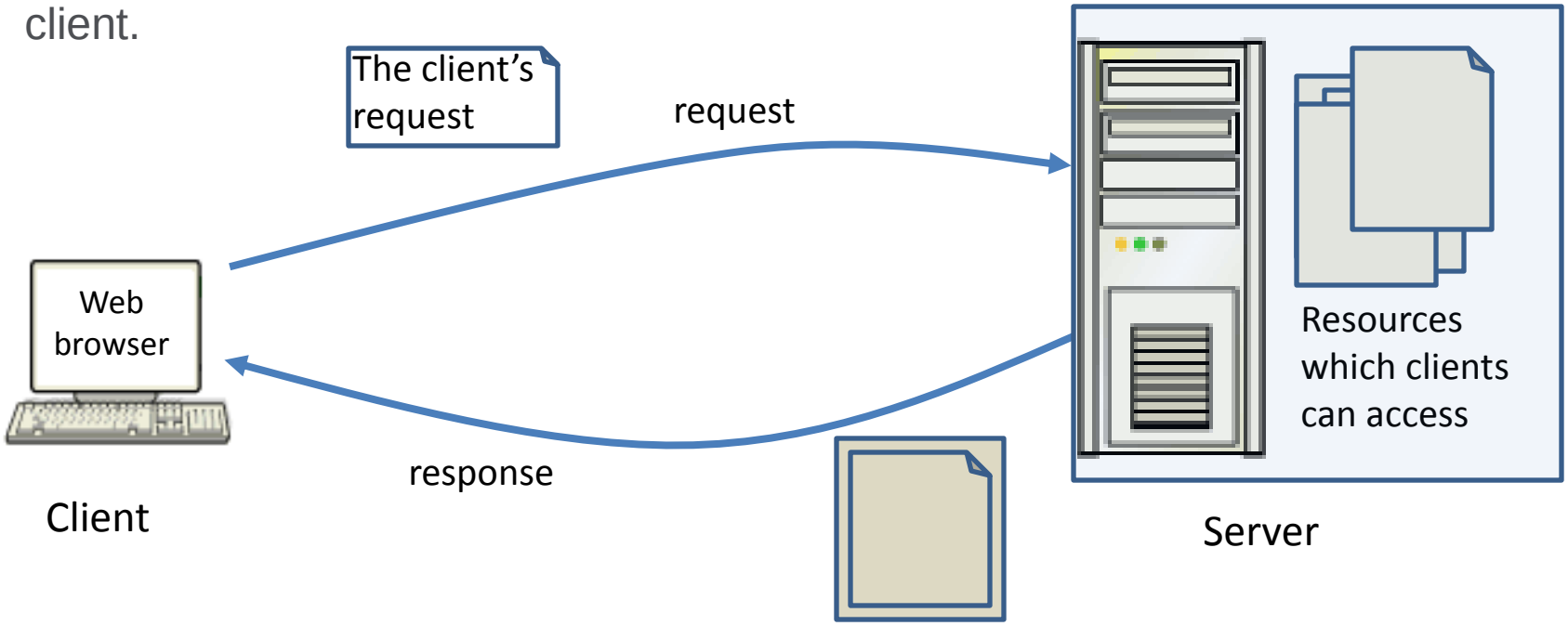
- **Web application** has a very general definition—client/server software that is connected by Internet technologies to route the data it processes.



Client, network, and server **-** shows the client/server relationship graphically

Web Server

- Web server, a server software, nearly always plays a central role in brokering communication (HTTP traffic) between client and server.
- A web server takes a client request and sends back the response to the client.



HTTP Client Server communication

- The language of the Web is the Hypertext Transfer Protocol (HTTP). HTTP is an application-level protocol that forms the basis of all directives we issue from our browsers (as well as many other standalone applications & software agents).

- HTTP functions as a request-response protocol in the client-server computing model.

- In HTTP, a web browser, for example, acts as a *client*, while an application running on a computer hosting a web site functions as a *server*.

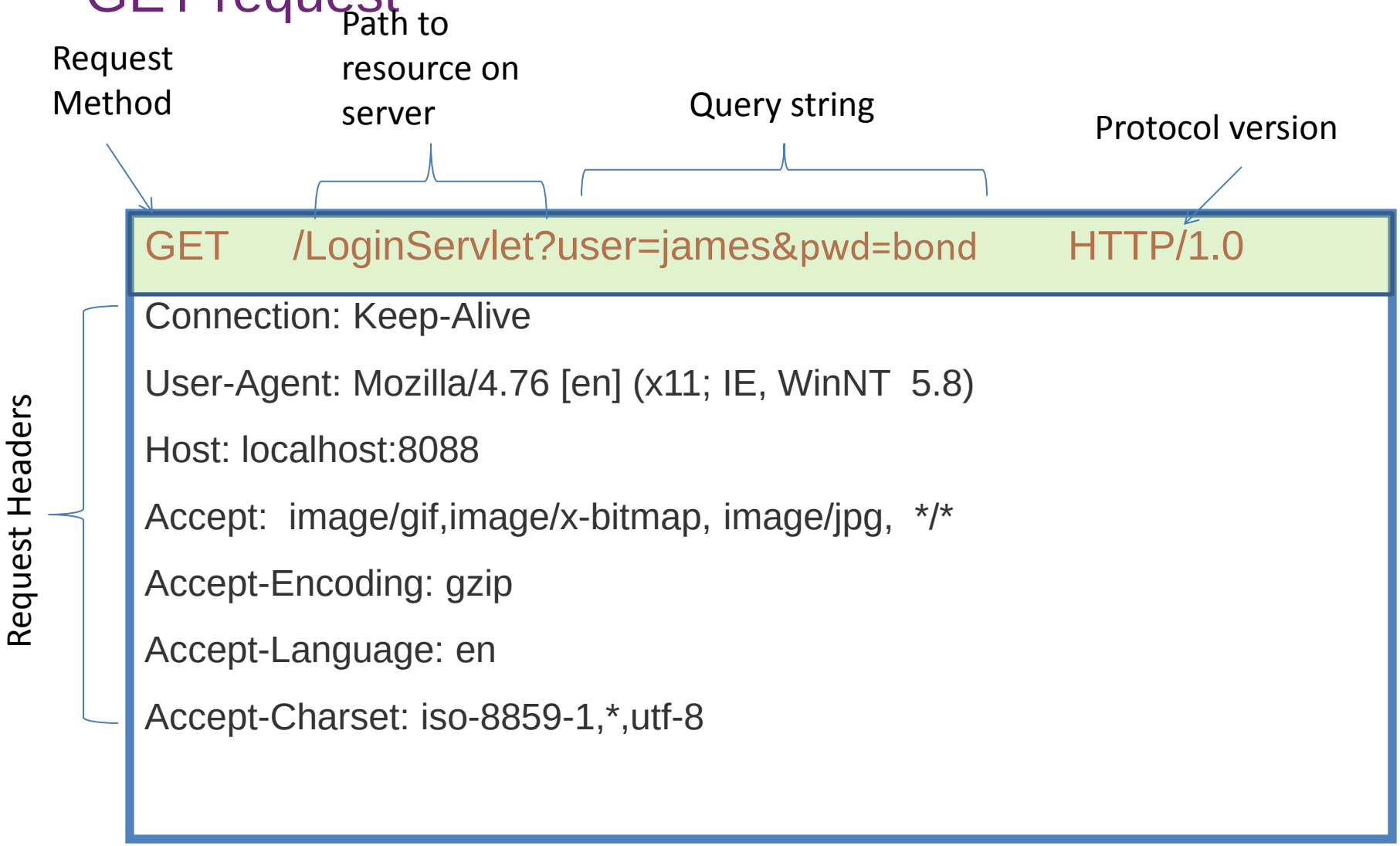
- The client submits an HTTP *request* message to the server. The server, which has *resources* such as HTML files, or performs other functions on behalf of the client, returns a response message to the client.

HTTP Request Methods

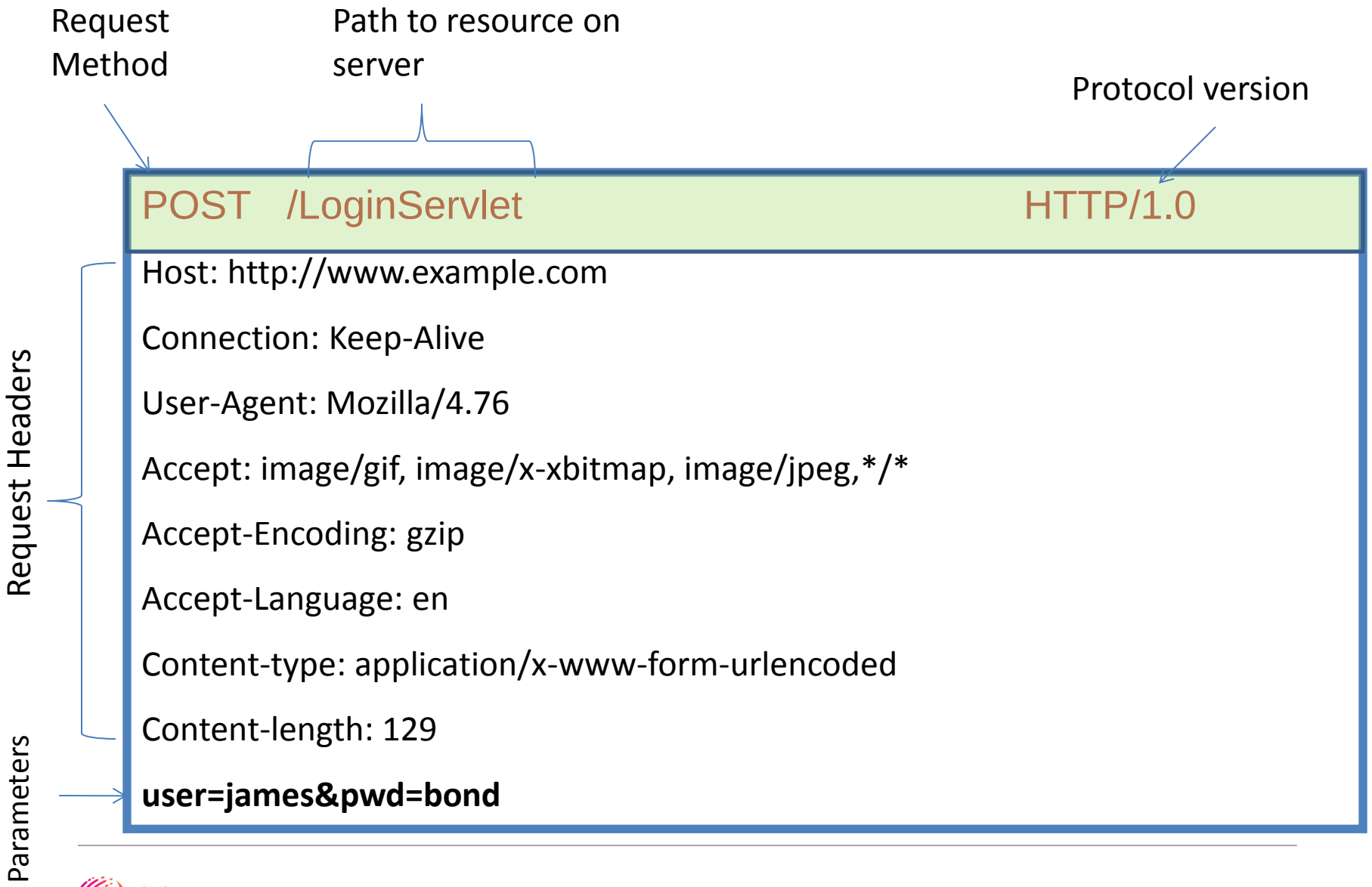
- **GET**
  - The HTTP GET method is used to retrieve the specified URI from the Web server.
  - Requests using GET "SHOULD NOT have the significance of taking an action other than retrieval".

- **POST**
  - Submits data to be processed (HTML form data) to the identified resource. The data is included in the body of the request. This may result in the creation of a new resource or the updates of existing resources or both.

GET request


Request Method
Path to resource on server
Query string
Protocol version
GET /LoginServlet?user=james&pwd=bond HTTP/1.0
Request Headers
Connection: Keep-Alive
User-Agent: Mozilla/4.76 [en] (x11; IE, WinNT 5.8)
Host: localhost:8088
Accept: image/gif,image/x-bitmap, image/jpg, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

POST request



HTTP Response

- After receiving and interpreting a request message, a server responds with an HTTP response message.

- A response from a Web server normally consists of a status line, one or more response headers, a blank line, and the document.

- Response =
  - Status-Line ;
  - header
  - CRLF
  - [message-body]

HTTP Response

Protocol version | Status Code | Reason-Phrase

HTTP/1.0 | 200 | OK

Header

Content-Type : text / html

Date: Friday 03 June 2011 15:27 GMT

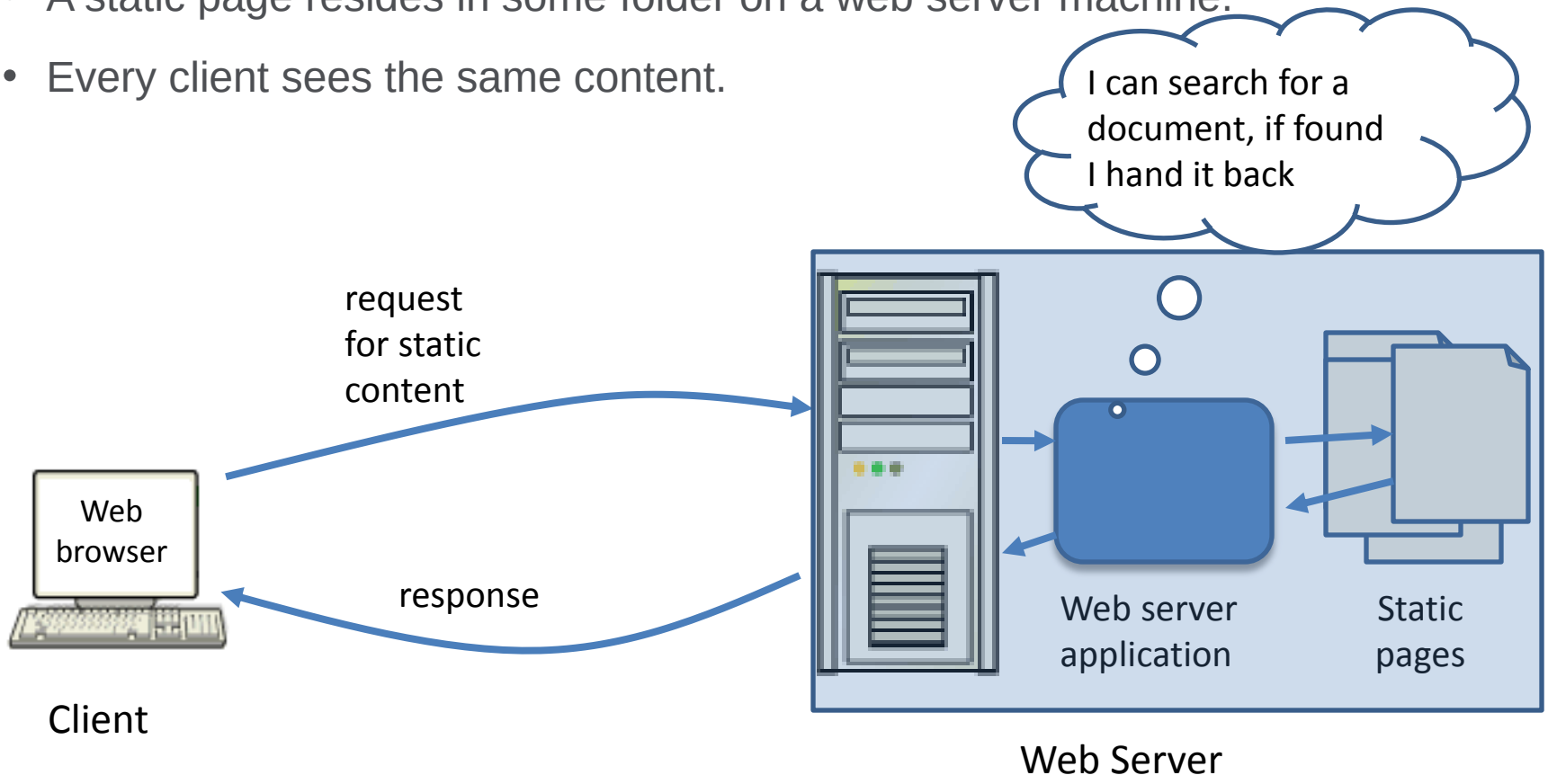
Server: Apache Tomcat.6.1

Body

```
<html>
        <body>
                <b> Hello World </b>
                <br/>
                Date in Server : Friday 03 June 2011 15:27
        </body>
</html>
```

Static Content

- Web server application by itself can just server static pages
- A static page resides in some folder on a web server machine.
- Every client sees the same content.



Dynamic Content

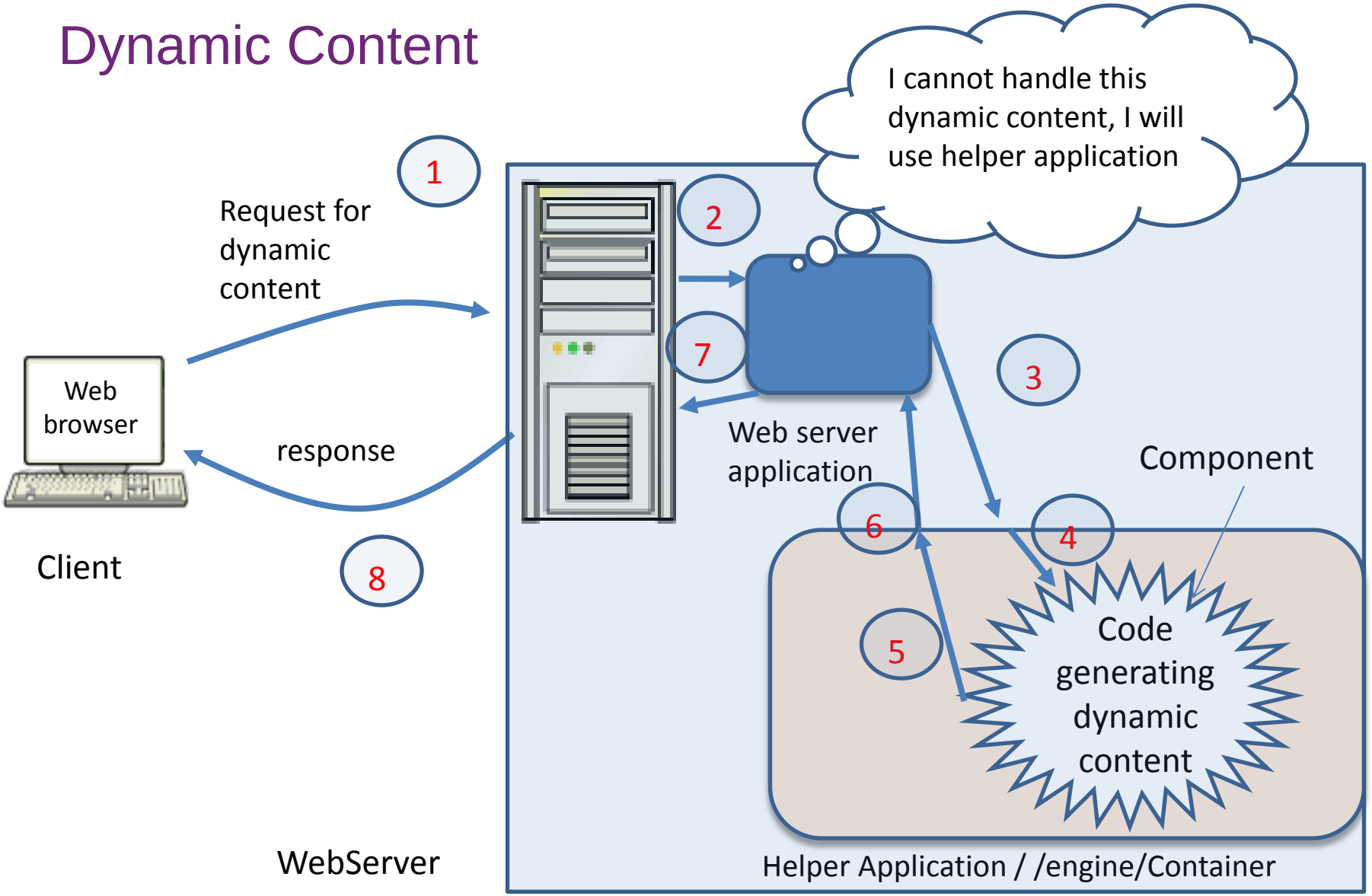
- Helper application on web server commonly known as engine/container are needed to generate dynamic content.

- If an helper application is ASP engine, the web server can serve dynamic contents using applications written using ASP.

- If an helper application is Servlet engine (Web Container) , then the web server can server dynamic content using Servlet/JSP technology.

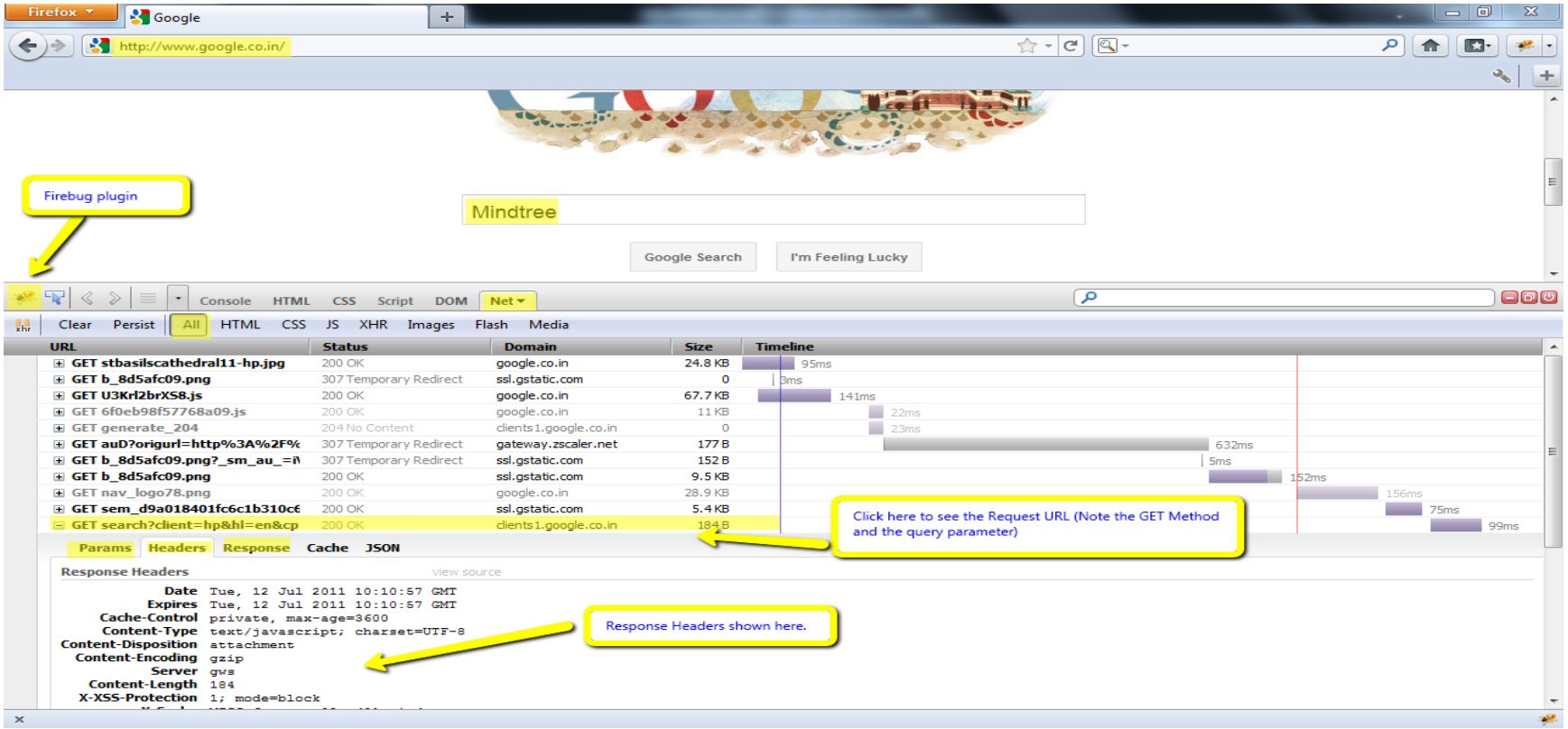
- Similarly for CGI, it can serve PERL, C , Python or Unix shell scripts.

Dynamic Content



Example

- Access www.google.com. Perform a search and observe the GET method & HTTP headers.



Servlet

- Servlet technology is used to build dynamic web component using Java programming language

- Servlet's executes within a Web Container.

- Servlet's receives the request and generates the response dynamically based on the request.

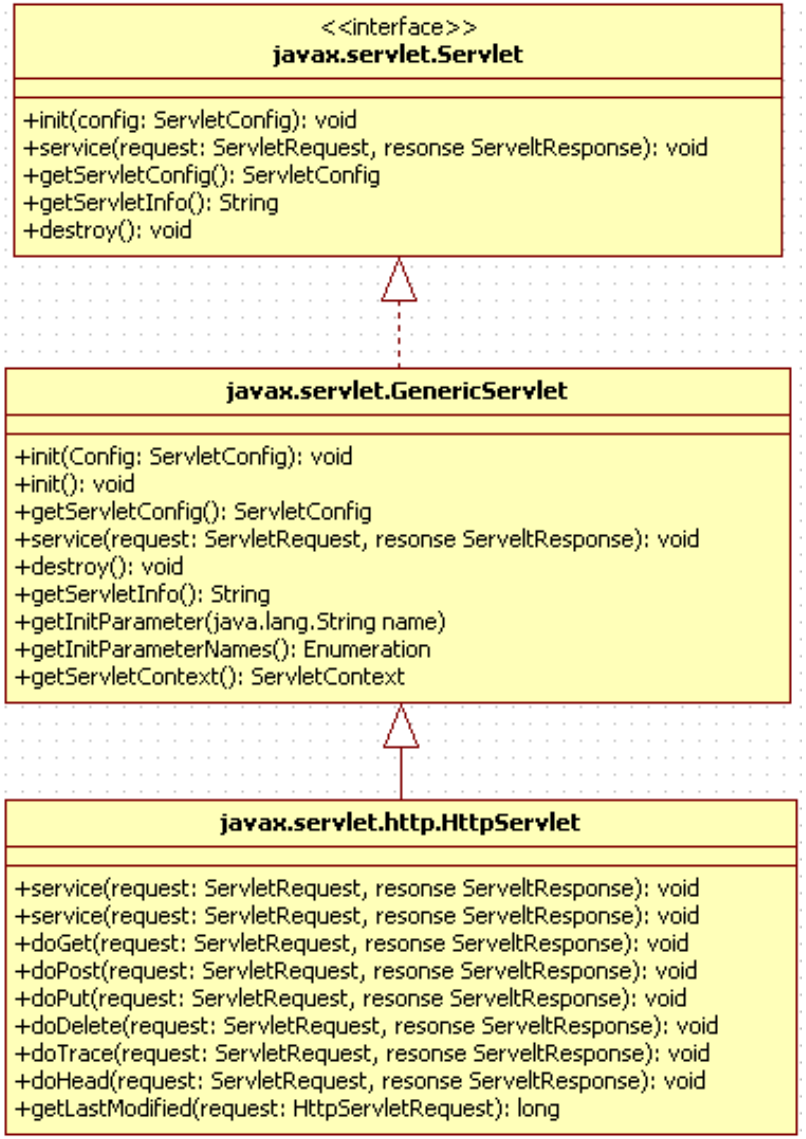
- Servlet's are not tied to a specific client-server protocol, but are most often used with the HTTP protocol. Therefore, the word "Servlet" is often used in the meaning of "HTTP Servlet".

Servlet

- Servlet's typically run on multithreaded servers, so be aware that a Servlet must handle concurrent requests and be careful to synchronize access to shared resources.

- Shared resources include in-memory data such as instance or class variables and external objects such as files, database connections, and network connections.

Servlet API


<<interface>>
javax.servlet.Servlet
+init(config: ServletConfig): void
+service(request: ServletRequest, resonse ServeltResponse): void
+getServletConfig(): ServletConfig
+getServletInfo(): String
+destroy(): void
javax.servlet.GenericServlet
+init(Config: ServletConfig): void
+init(): void
+getServletConfig(): ServletConfig
+service(request: ServletRequest, resonse ServeltResponse): void
+destroy(): void
+getServletInfo(): String
+getInitParameter(java.lang.String name)
+getInitParameterNames(): Enumeration
+getServletContext(): ServletContext
javax.servlet.http.HttpServlet
+service(request: ServletRequest, resonse ServeltResponse): void
+service(request: ServletRequest, resonse ServeltResponse): void
+doGet(request: ServletRequest, resonse ServeltResponse): void
+doPost(request: ServletRequest, resonse ServeltResponse): void
+doPut(request: ServletRequest, resonse ServeltResponse): void
+doDelete(request: ServletRequest, resonse ServeltResponse): void
+doTrace(request: ServletRequest, resonse ServeltResponse): void
+doHead(request: ServletRequest, resonse ServeltResponse): void
+getLastModified(request: HttpServletRequest): long

First Servlet

```
/**
 * @author Banu Prakash
 * © 2011 MindTree Limited
 * Servlet implementation class SampleServlet
 */
public class SampleServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
     *      response)
     */
    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.print("<html><body>");
        out.print("Hello World");
        out.print("<br /> Time in Server : " + new Date());
        out.print("</body></html>");
    }
}
```

The Deployment Descriptor

- A **deployment descriptor** (DD) refers to a configuration file for an object that is deployed to a container.
- XML is used for the syntax of these deployment descriptors.
- The web.xml is the deployment descriptor for web applications.
  - File should reside in WEB-INF folder
- The persistence.xml is the deployment descriptor for Java Persistence API
  - File should reside in META-INF folder
- The application.xml is the deployment descriptor for complete Enterprise application
  - File should reside in META-INF folder

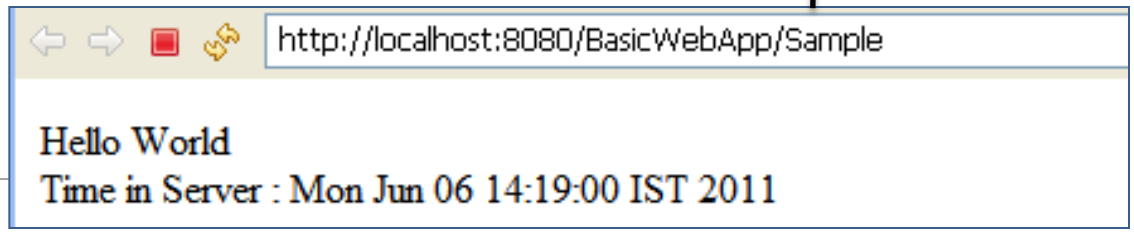
The web.xml

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    id="WebApp_ID" version="2.5">


  <servlet>
    <servlet-name>SampleServlet</servlet-name>
    <servlet-class>com.mindtree.web.SampleServlet</servlet-class>
  </servlet>


  <servlet-mapping>
    <servlet-name>SampleServlet</servlet-name>
    <url-pattern>/Sample</url-pattern>
  </servlet-mapping>


</web-app>
```

<servlet-name>
The name of the servlet, used to reference the servlet definition elsewhere in the DD

<servlet-class>
The fully-qualified class name of the servlet.

<servlet-mapping>
The servlet-mapping element defines a mapping between a servlet and a URL pattern

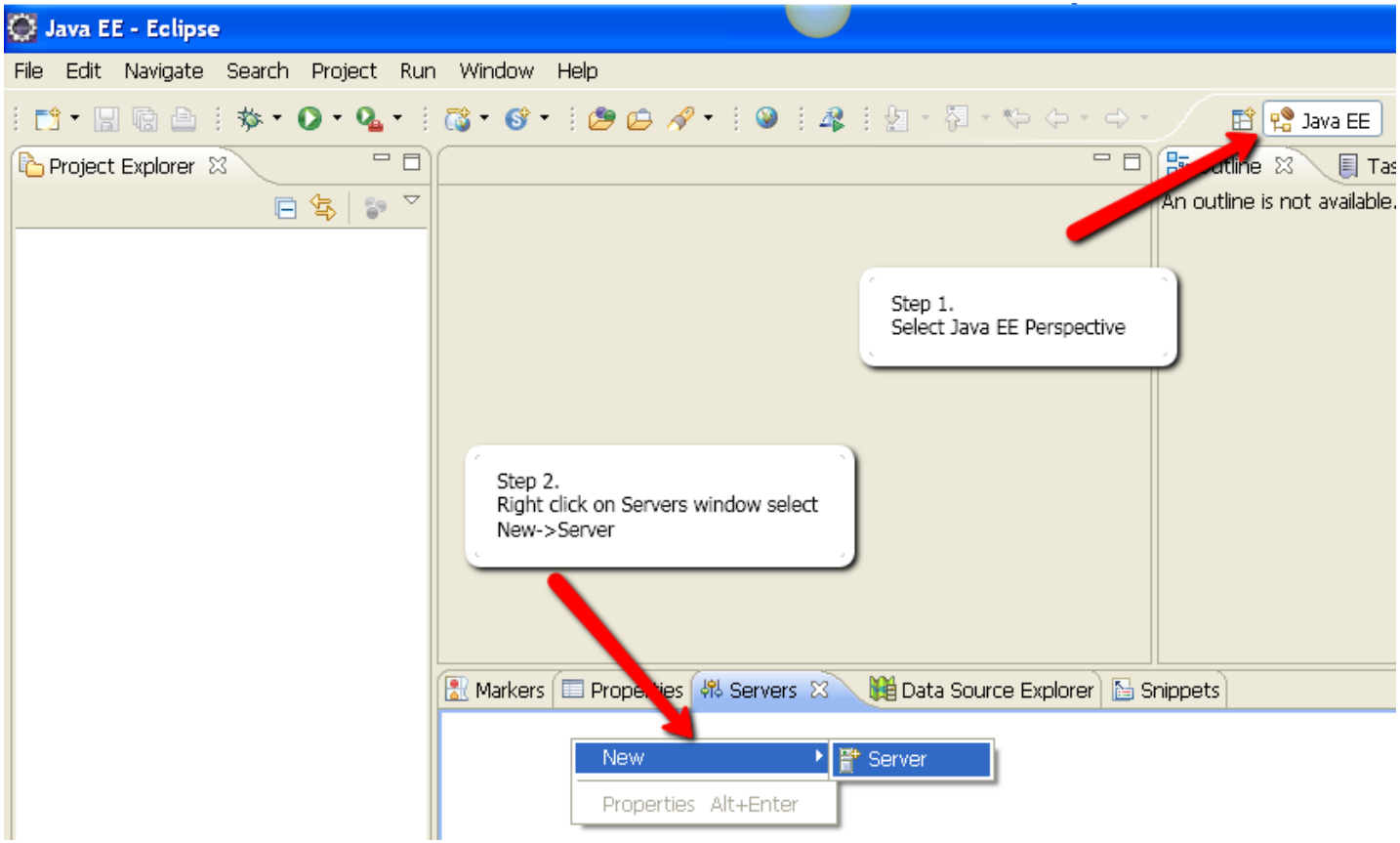


Commonly Overridden HttpServlet Methods

Method	Purpose
void init()	Convenience method used during servlet instantiation . Any session-independent data structures (such as a cache or creating database connection pools) can be initialized here.
void doGet(HttpServletRequest a_req, HttpServletResponse a_resp)	Processes HTTP GET requests. The details of the request are in a_req and the output stream for formatting the response can be obtained using a_resp.
void doPost(HttpServletRequest a_req, HttpServletResponse a_resp)	Processes HTTP POST requests. The details of the request are in a_req and the output stream for formatting the response can be obtained using a_resp.
void destroy()	Convenience method used during servlet destruction. Any session-independent data structures can be cleaned up here. (connections closed, etc.)

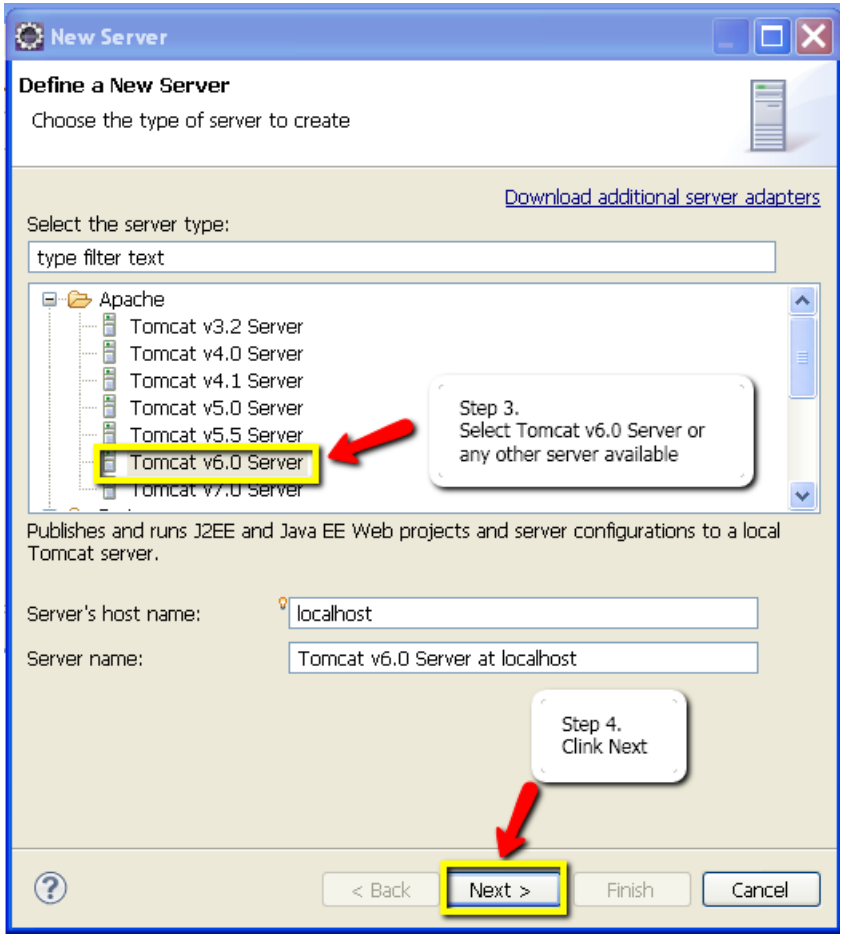
Build our first dynamic web application using eclipse

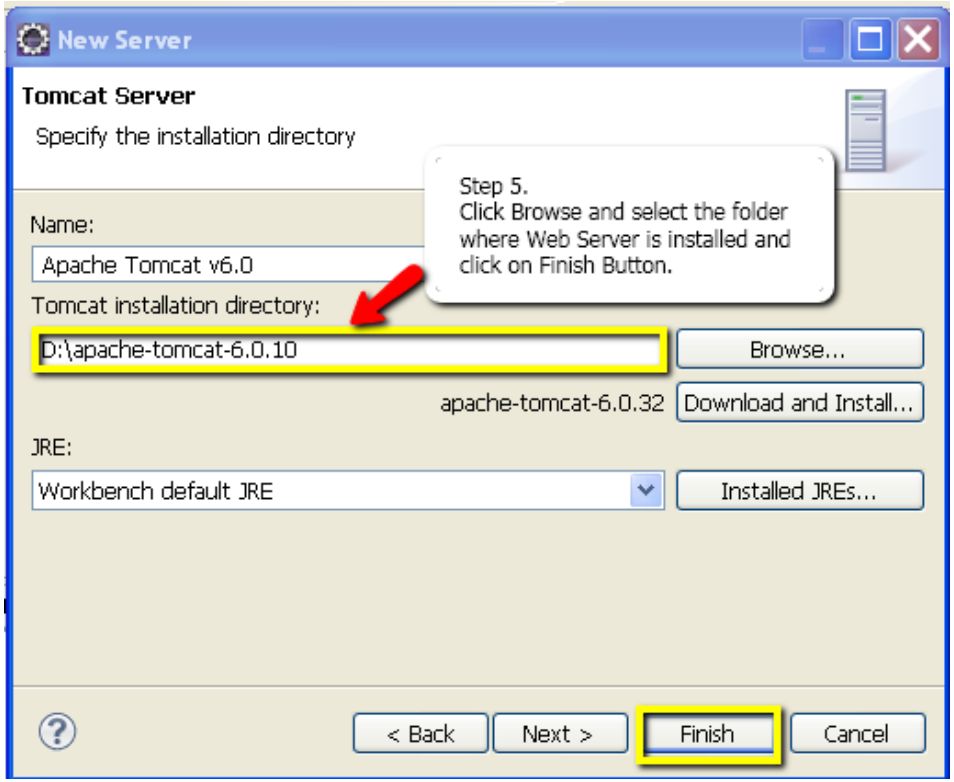
- Select JEE Perspective and add New Server



Build our first dynamic web application using eclipse

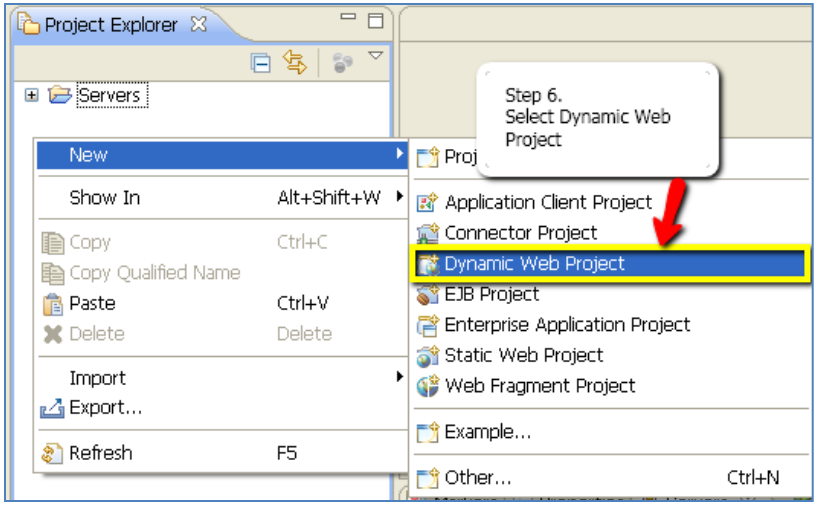
- Select the Server on which you need to deploy your web application

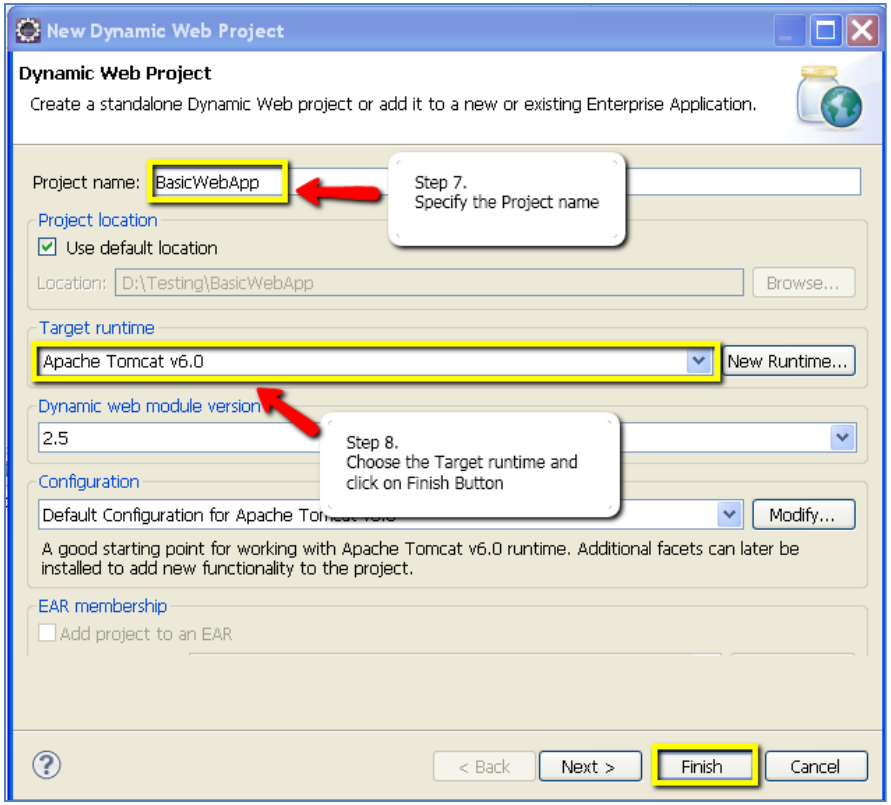




Build our first dynamic web application using eclipse

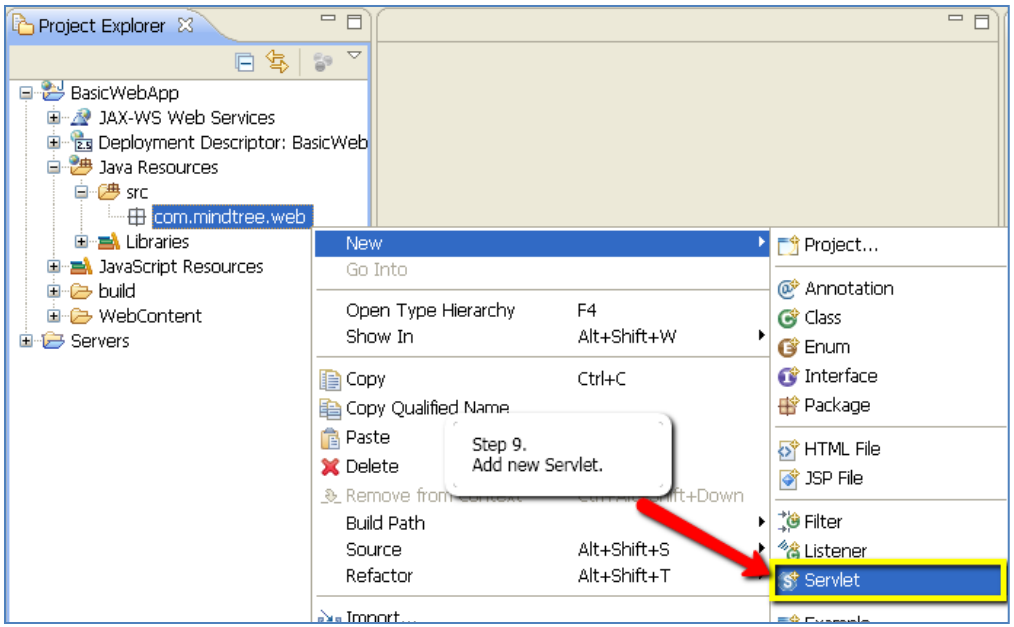
- Add a new Dynamic Web Project to workspace
- Specify the Target run time for the dynamic web project

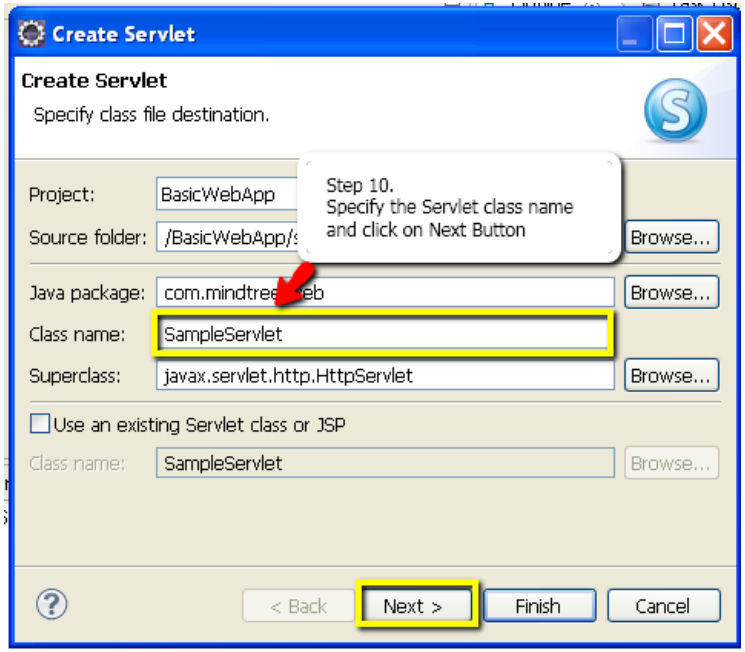




Build our first dynamic web application using eclipse

- Writing your First Servlet
  - Add a new Servlet to the dynamic web project
  - Specify the Servlet class name



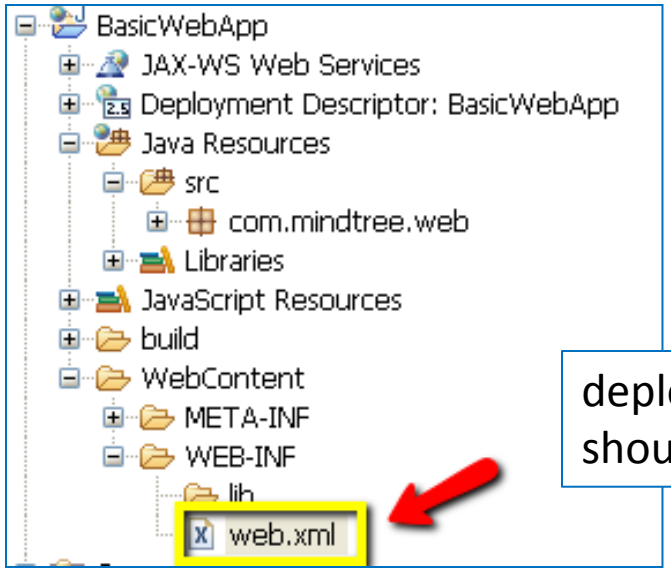


Coding your first Servlet: SampleServlet

```java
/**
 * @author Banu Prakash
 * © 2011 MindTree Limited
 * Servlet implementation class SampleServlet
 */
public class SampleServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
     *      response)
     */
    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.print("<html><body>");
        out.print("Hello World");
        out.print("<br /> Time in Server : " + new Date());
        out.print("</body></html>");
    }
}
```

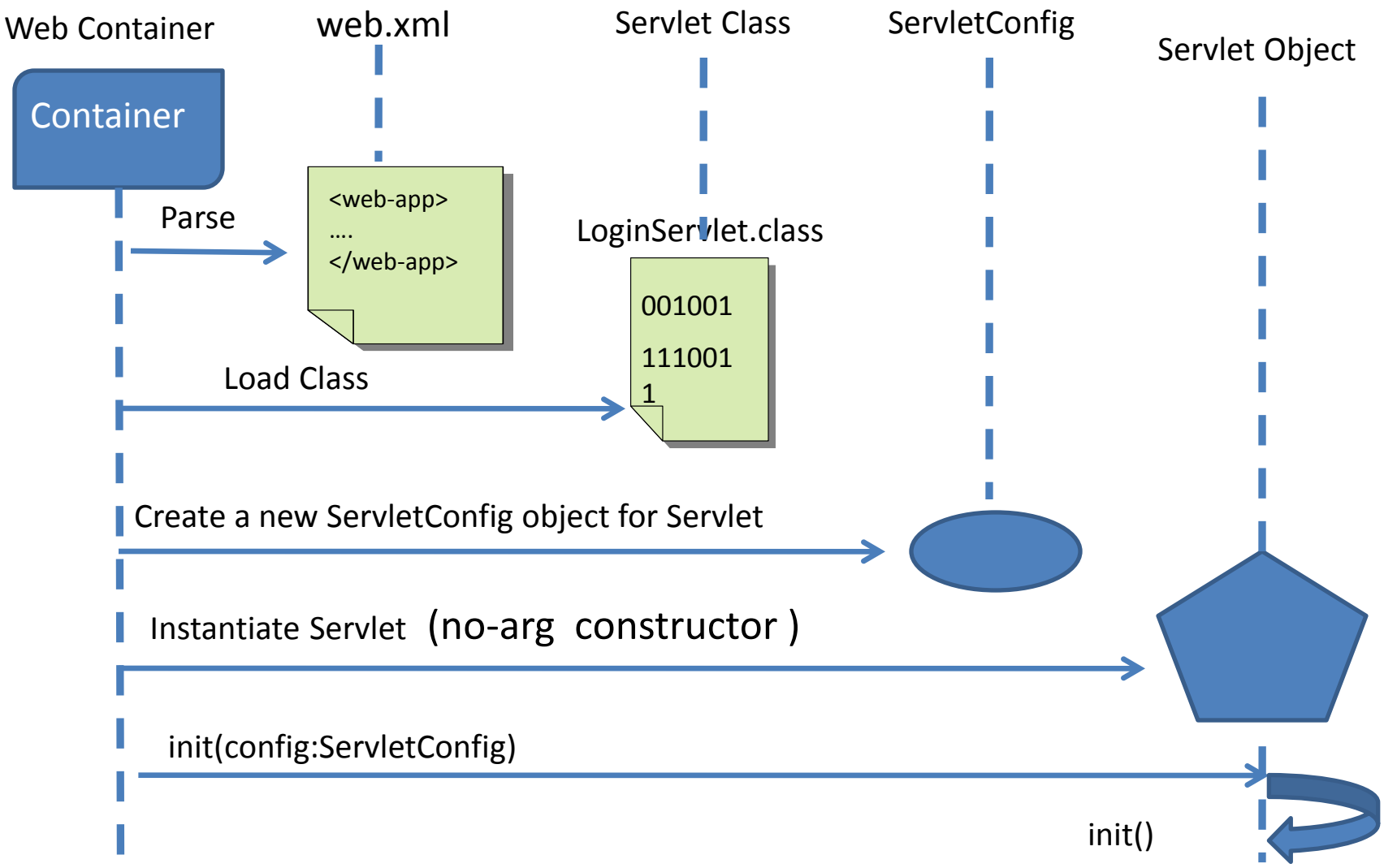
Deploying your web application

- Deployment Descriptor
  - A **deployment descriptor** (DD) refers to a configuration file for an object that is deployed to a container.
  - XML is used for the syntax of these deployment descriptors.
  - The web.xml is the deployment descriptor for web applications.

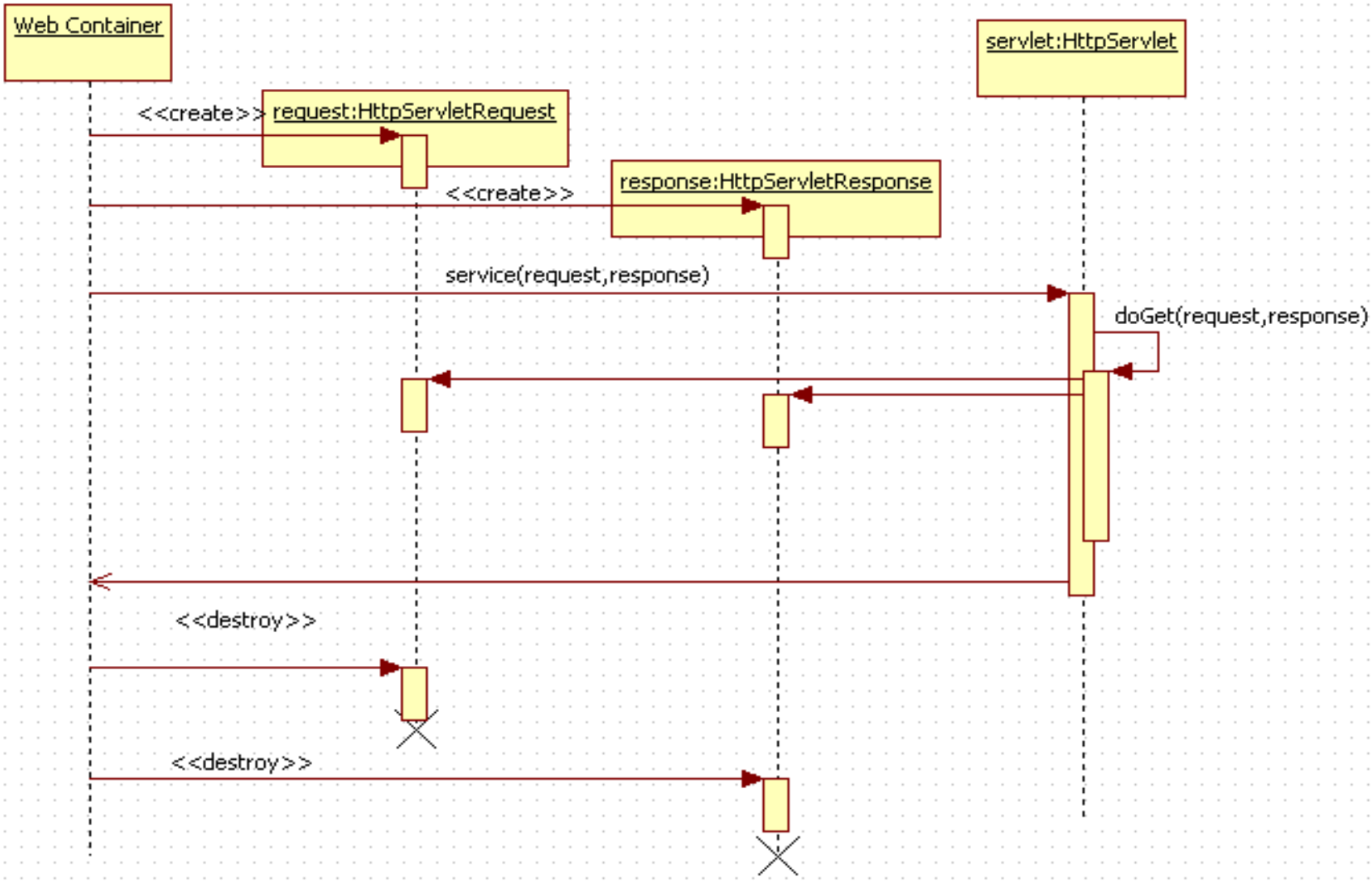


deployment descriptor “web.xml” should be under WEB-INF folder

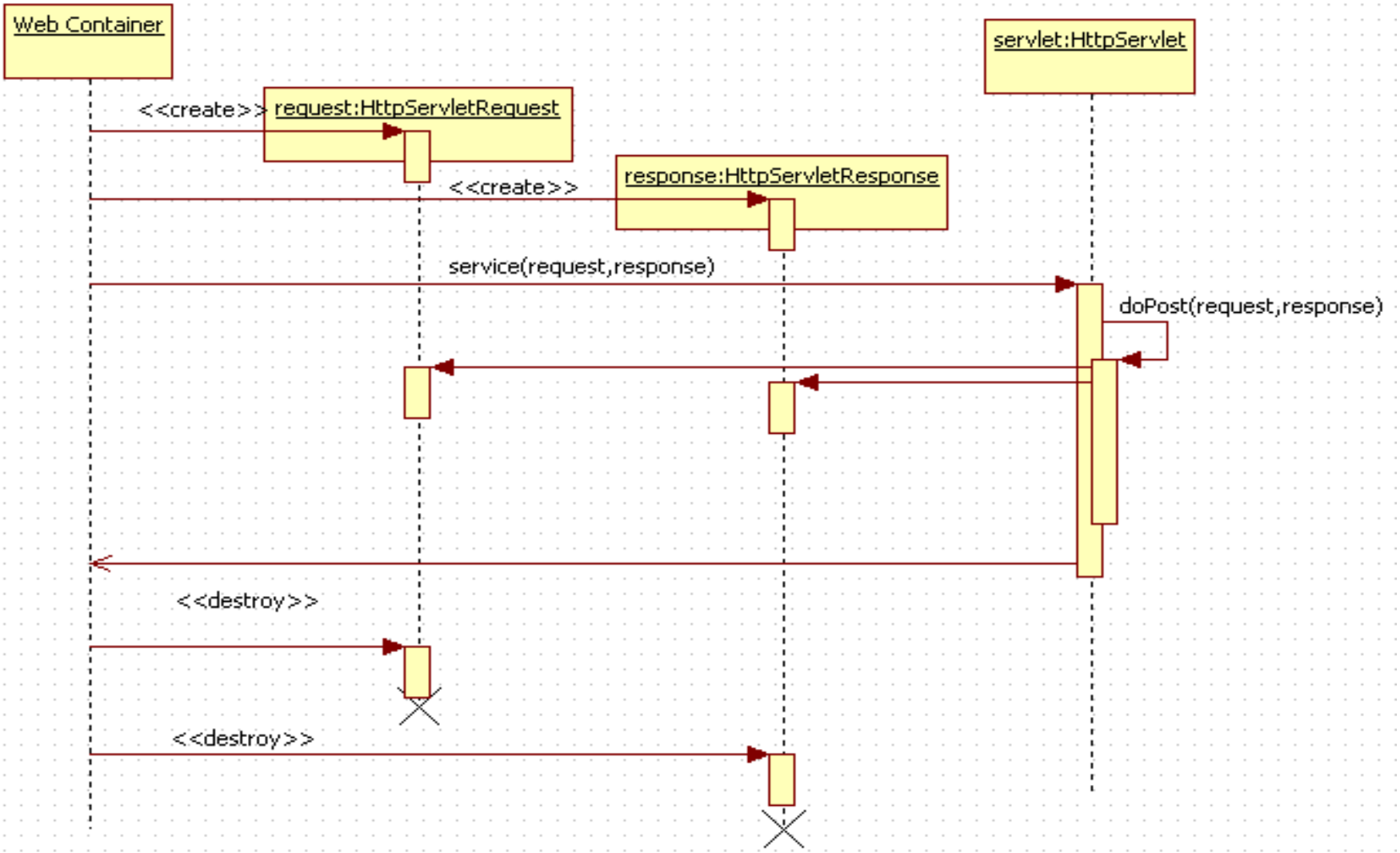
Servlet's Deployment



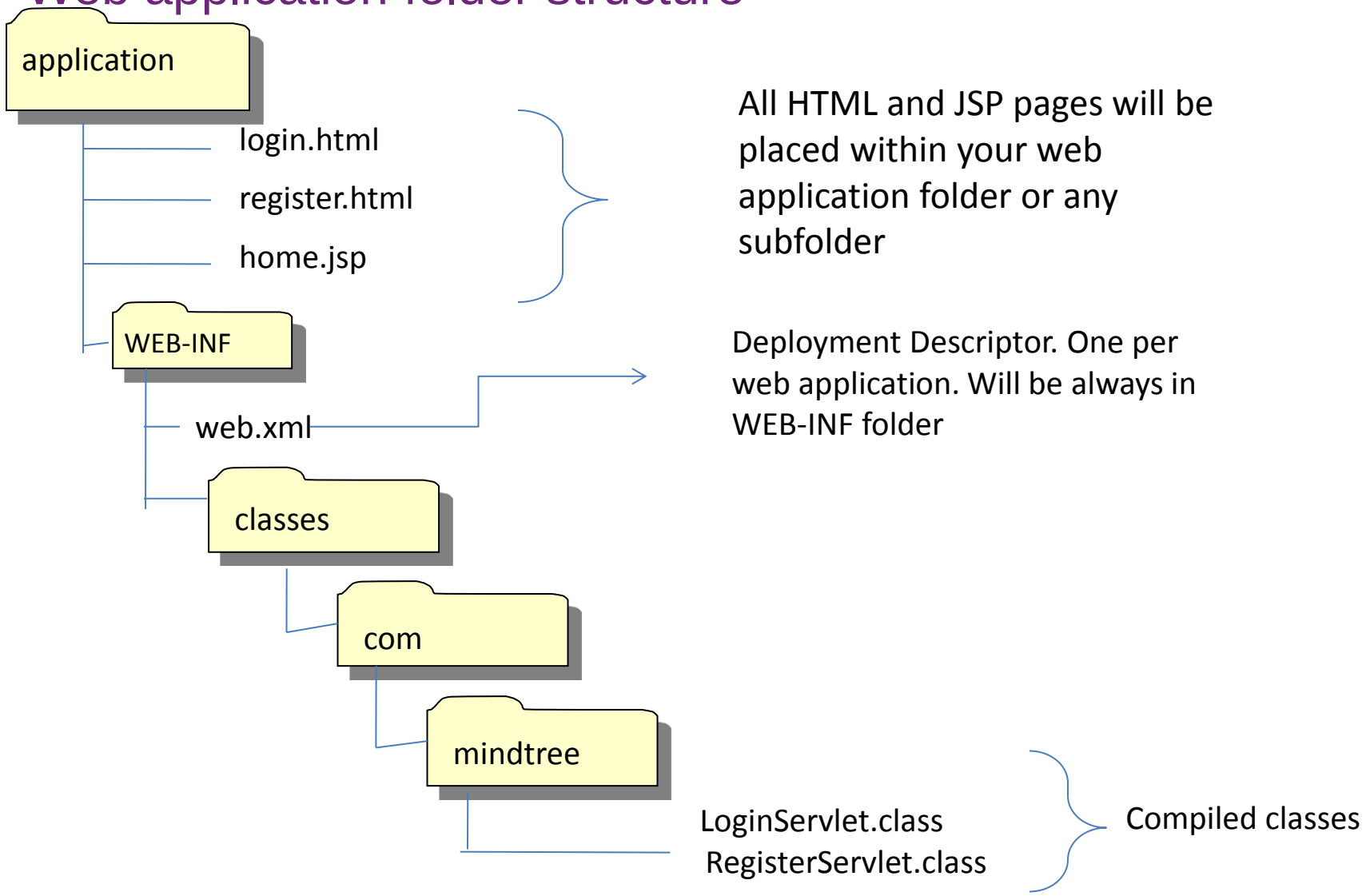
Sequence diagram for HTTP GET request



Sequence diagram for HTTP POST request



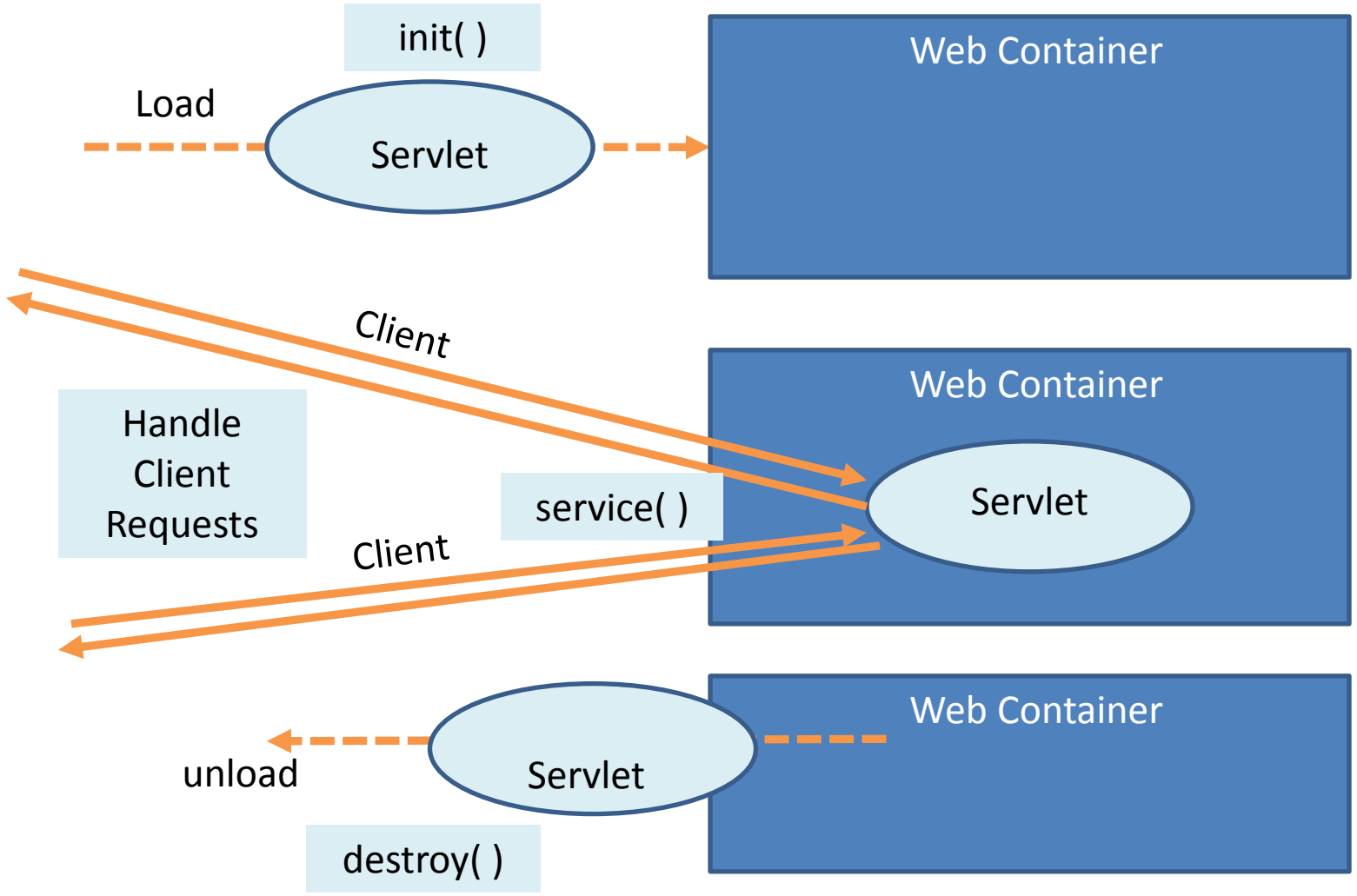
Web application folder structure



All HTML and JSP pages will be placed within your web application folder or any subfolder

Deployment Descriptor. One per web application. Will be always in WEB-INF folder

Servlet's Life-Cycle



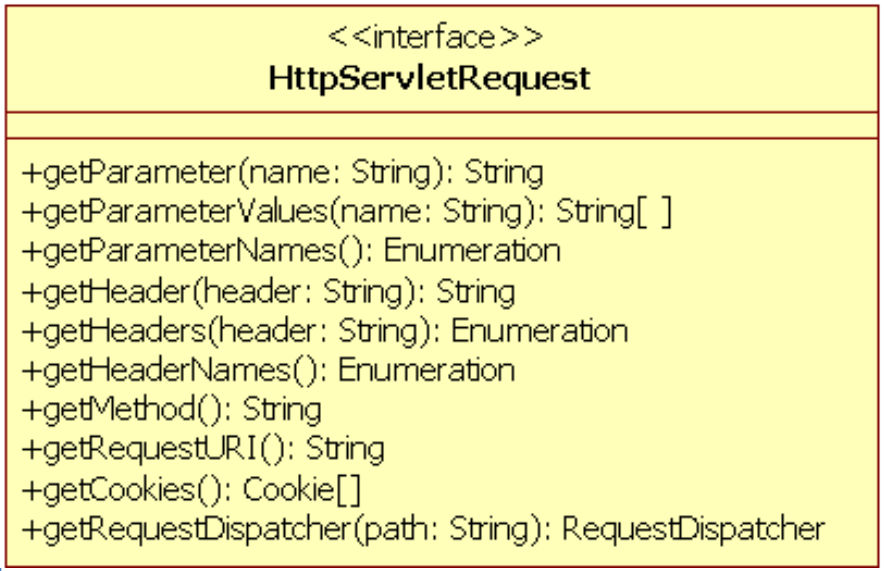
HttpServletRequest

- An HttpServletRequest object provides access to HTTP header data, and arguments that the client sent as part of the request.

String **getParameter**(String name) Returns the value of a request parameter as a String, or null if the parameter does not exist.

String[ ] **getParameterValues**(String name) Returns an array of String objects containing all of the values for the given request parameter , or null if the parameter does not exist.

public String **getHeader**(String name) Returns the value of the specified request header as a String.



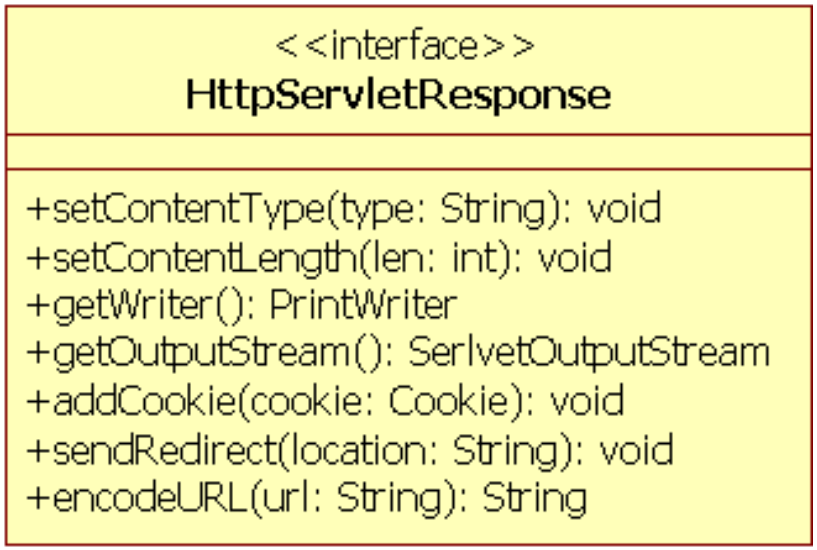
HttpServletResponse

- HttpServletResponse object assists a servlet in sending a response to the client.

•ServletOutputStream getOutputStream().
This method returns ServletOutputStream to send binary data in a body response.

•PrintWriter getWriter( )
This method returns Printriter to send character data,.

•void sendRedirect (String location)
Sends a temporary redirect response to the client using the specified redirect location URL and clears the buffer



Mindtree

Welcome to possible

References

Contains the reference that will supplement the self learning and will be needed for completing the assignments & practice questions

References

- Series of Servlet tutorials:
  - http://www.servletworld.com/servlet-tutorials/j2ee-web-development-overview.html
- Fundamentals of Servlets
  - http://java.sun.com/developer/onlineTraining/Servlets/Fundamentals/servlets.html
- SingleThreadModel
  - http://www.google.co.in/search?sourceid=navclient&aq=0h&oq=Single&ie=UTF-8&rlz=1T4SKPT_enIN417IN420&q=single+thread+model+in+servlets
- Sending Binary data
  - http://www.novocode.com/doc/servlet-essentials/chapter2d.html

Welcome to possible

Name

Email

www.mindtree.com/social