Introduction to Web Container

What is HTTP?

- HTTP →Hyper-Text-Transfer-Protocol
- Stateless and application-level protocol
- the mode by which the web pages are communicated across the internet.
- globally accepted method.

How does it work?

•Suppose a user browsing the web site
www.facebook.com/index.html
•The browser makes a connection with the server www.facebook.com by finding its IP address from a DNS server.
•The browser then sends a request to server to fetch the file /index.html.
•The server responds with information about the page and the contents of the HTML page itself.

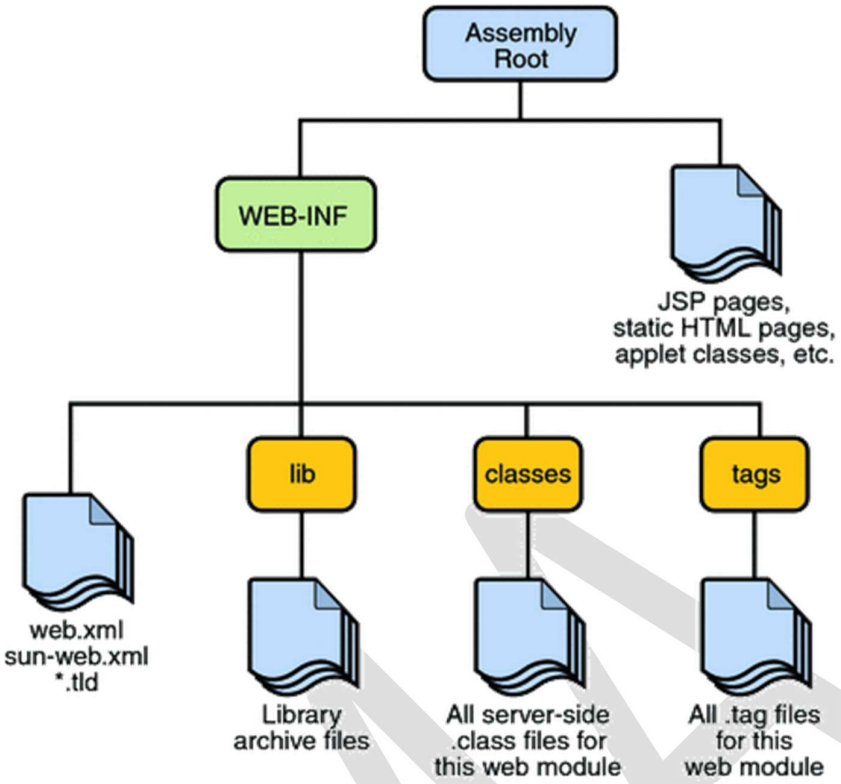
HTTP Requests

Methods
•GET -retrieves the resource identified by request URL
•HEAD -returns the headers
•POST -sends the data of unlimited length to Web Server
•PUT -stores a resource under the request URL
•DELETE –deletes the resource identified by request URL
•OPTIONS –returns the HTTP methods the server supports
•TRACE -returns the header fields sent with the TRACE request

HTTP Response

To indicate the type of content in request and response bodies, they use **M**ulti-purpose **I**nternet **M**ail **E**xtensions (MIME).
MIME types include text/html and image/gif.

- Web Application is a collection of :
  - **Web components** (Servletsand JSP's)
  - Static resource files such as images
  - Helper classes
  - Libraries
  - Deployment descriptor (web.xml file)
- Web Application can be represented as
  - A hierarchy of directories and files (**unpacked form**) or
  - *.WAR file reflecting the same hierarchy (**packed form**)



Web container

A Web container serves as a runtime environment for a Web application.
(Naming context and life-cycle management)
The web application runs within a Web container of a Webserver.
A Web container may communicate with other Web containers.

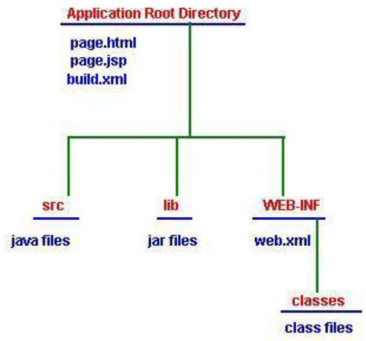
Web Application Development and Deployment Steps

1.Write (and compile) the Web component code (Servlet or JSP) and helper classes referenced by the web component code
2.Create any static resources (for example, images or HTML pages)
3.Create deployment descriptor (web.xml)
4.Build the Web application (*.war file or deployment-ready directory)
5.Deploy the web application into a Web container
*Web clients are now ready to access them via URL

1. Write and compile the Web component code

- Create development tree structure
- Write either servlet code or JSP pages along with related helper code
- Create build.xml for Ant-based build (and other application development life-cycle management) process
- IDE (i.e. NetBeans, Eclipse) handles all these chores

Develop tree structure



2. Create any static resources

HTML pages
– Custom pages
– Login pages
– Error pages
Image files that are used by HTML pages or JSP pages
– Example: duke.waving.gif 

3. Create deployment descriptor (web.xml)

? Deployment descriptor contains deployment time runtime instructions to the Web container
– URL that the client uses to access the web component
? Every web application has to have it

4. Build the Web application

? Either *.WAR file or unpacked form of *.WAR file
? Build process is made of
– create build directory (if it is not present) and its subdirectories
– compile Java code into build/WEB-INF/classes directory
? Java classes reside under ./WEB-INF/classes directory
– copy web.xml file into build/WEB-INF directory
– copy image files into build directory

5. Deploy Web application

? Deploy the application over deployment platform such as Sun Java System App Server or Tomcat
? 3 ways to deploy to Sun Java System App server
– asadmin deploy --port 4848 --host localhost –
passwordfile "c:\j2eetutorial14\examples\common\adminpassword.
txt" --user admin hello2.war (asant deploy-war)
– App server admin console
– NetBeans

6. Perform Client Access to Web Application

? From a browser, go to URL of the Web application

Configuring Web Application via web.xml

Web Applications Deployment Descriptor (web.xml)

•helps in managing the deployment configuration of web applications.
•stored in the /WEB-INF directory of the application.
•purposes:
•Initialization of parameters for Servlets and we bapplications
•Servlet/JSP definitions
•Servlet/JSP mappings
•MIME types (**M**ulti-purpose **I**nternet **M**ail **E**xtensions)
•Security

? Prolog
? Alias Paths
? Context and Initialization Parameters
? Event Listeners
? Filter Mappings
? Error Mappings
? Reference to Environment Entries, Resource environment entries, or Resources
? Case sensitive
? Order sensitive (in the following order)
– icon, display-name, description, distributable
– context-param, filter, filter-mapping
– listener, servet, servlet-mapping, session-config
– mime-mapping, welcome-file-list
– error-page, taglib, resource-env-ref, resource-ref
– security-constraint, login-config, security-role
– env-entry, ejb-ref, ejb-local-ref

Prolog

? Every XML document needs a prolog

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/dtd/webapp_2_3.dtd">
```

Alias Paths (of web.xml)

? When a request is received by Servlet container, it must determine which Web component in a which web application should handle the request. It does so by mapping the URL path contained in the request to a Web component

? A URL path contains the context root and alias path

– http://<host>:8080/context_root/alias_path

? Alias Path can be in the form of either

– /alias-string (for servlet) or

– /*.jsp (for JSP)

```
<web-app>
    <welcome-file-list>
        <welcome-file>test.html</welcome-file>
        </welcome-file-list>
    <servlet>
        <servlet-name>MyServlet</servlet-name>
        <servlet-class>MyServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>MyServlet</servlet-name>
        <url-pattern>/servlet1</url-pattern>
    </servlet-mapping>
</web-app>
```

Context and Initialization Parameters (of web.xml)

? Represents application context

? Can be shared among Web components in a WAR file

```
<web-app>
...
<context-param>
<param-name>
javax.servlet.jsp.jstl.fmt.localizationContext
</param-name>
<param-value>messages.BookstoreMessages</param-value>
</context-param>
...
</web-app>
```

Event Listeners (of web.xml)

? Receives servlet life-cycle events

```
<listener>
<listener-class>listeners.ContextListener</listener-class>
</listener>
```

Filter Mappings (of web.xml)

? Specify which filters are applied to a request, and in what order

```
<filter>
<filter-name>OrderFilter<filter-name>
<filter-class>filters.OrderFilter<filter-class>
</filter>
<filter-mapping>
<filter-name>OrderFilter</filter-name>
<url-pattern>/receipt</url-pattern>
</filter-mapping>
```

Error Mappings (of web.xml)

? Maps status code retURLed in an HTTP response to a Java programming language exception retURLed by any Web component and a Web resource

<error-page>
<exception-type>exception.OrderException</exception-type>
<location>/errorpage.html</location>
</error-page>

References (of web.xml)

? Need when web components make references to environment entries, resource environment entries, or resources such as databases

? Example: declare a reference to the data source

<resource-ref>
<res-ref-name>jdbc/BookDB</res-ref-name>
<res-type>javax.sql.DataSource</res-type>
<res-auth>Container</res-auth>
</resource-ref>

Understanding Servlet Programming

Introduction to Servlet

- Servlet is the basic building blocks for building web-based interfaces to applications.
- Servlet Technology is provided by Sun.
- Applet: a java program that runs within the web browser.
- Servlet: a java program that runs within the web server.

    - A Servlet is a Java programming language class used to extend the capabilities of Java-enabled Web servers.
    - Servlets can respond to any type of request.
    - Servlets are protocol- and platform independent server-side components.
    - Interfaces and classes are provided by two packages:
        - javax.servlet
        - javax.servlet.http

What can you build with servlets?

- Search Engines
- Personalization Systems
- e-Commerce Applications
- Shopping Carts
- Product Catalogs
- Intranet Applications
- Groupware Applications: bulletin boards, file sharing, etc.

Advantages

- Servlets have six main advantages:
    - Efficient
    - Convenient
    - Powerful
    - Portable
    - Secure
    - Inexpensive

Packages java.servlet

- The package contains a number of classes and interfaces.
- The central abstraction of the servlet API is ***Servlet*** interface.
- Two classes **GenericServlet** and **HttpServlet** implement this interface.
- The package contains fourteen interfaces:
    - The servlet container provides the implementation of seven interfaces.
    - The programmer building the Web application implements the remaining seven interfaces.

Interfaces by Servlet container

- ServletConfig
- ServletContext
- ServletRequest
- ServletResponse
- RequestDispatcher
- FilterChain
- FilterConfig

Interfaces by Programmer

1. Servlet
2. ServletRequestListener
3. ServletRequestAttributeListener
4. ServletContextListener
5. ServletContextAttributeListener
6. SingleThreadModel
7. Filter

Exception

ServletException
UnavailableException

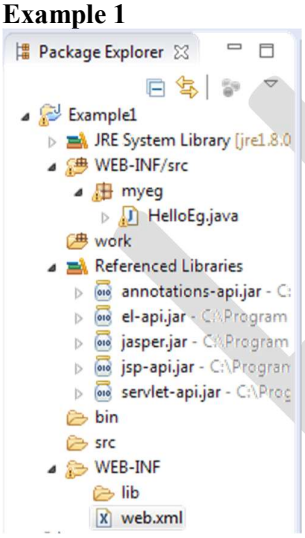
What is J2EE?

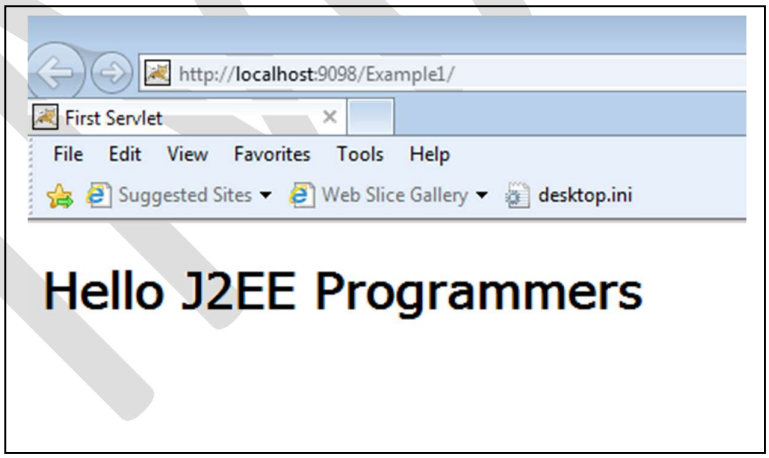
- J2EE / Java EE is a platform popularly used for
  - building server-side applications, and
  - provides a convenient component-based approach.
- It is a distributed computing architecture.
- Java EE is targeted at large-scale business systems.

Need for Enterprise Programming

- Enterprise means a business organization.
- Enterpriseapplicationsarethosesoftwareapplicationsthatfacilitatevariousactivitiesinanenterprise.
- Enterprise software is an integral part of a Information System, and as such includes web site software production.
- E.g.,
  - online shopping system
  - online payment processing system

Example 1

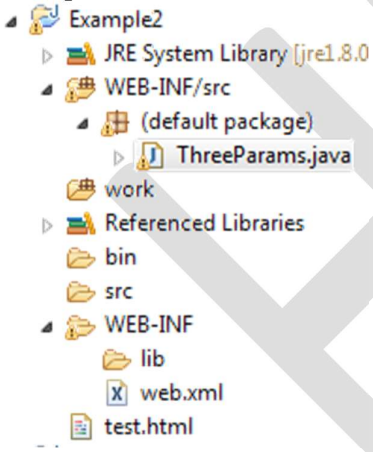


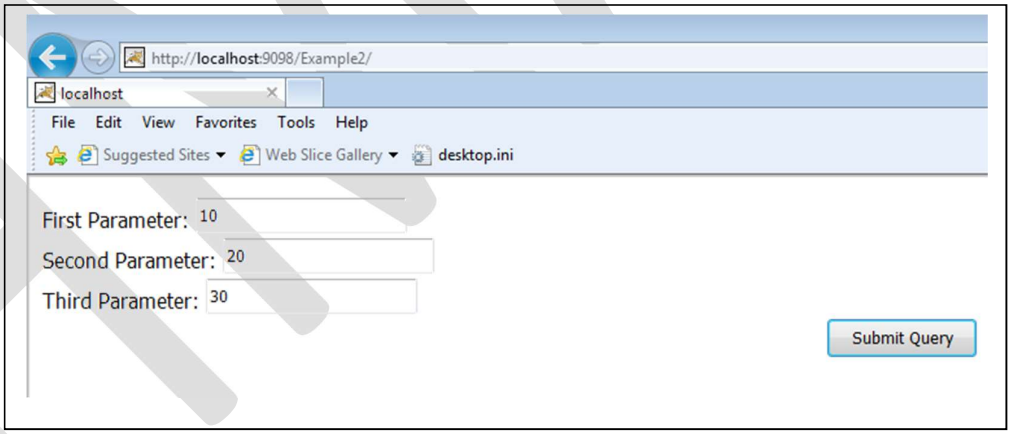


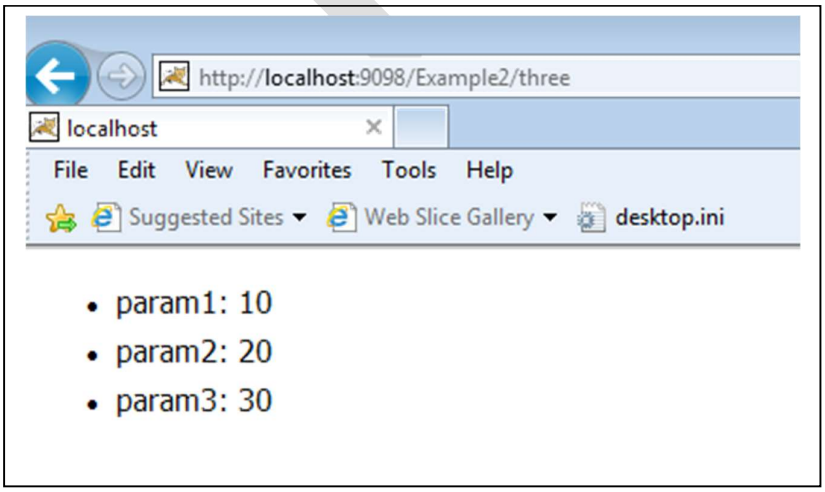

```
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
        metadata-complete="true">
      <welcome-file-list>
            <welcome-file>hello</welcome-file>
      </welcome-file-list>
      <servlet>
            <servlet-name>HelloEg</servlet-name>
            <servlet-class>myeg.HelloEg</servlet-class>
      </servlet>
      <servlet-mapping>
            <servlet-name>HelloEg</servlet-name>
            <url-pattern>/hello</url-pattern>
      </servlet-mapping>
</web-app>
```

```
package myeg;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class HelloEg extends HttpServlet {
      public  void  doGet(HttpServletRequest  request,  HttpServletResponse  response)  throws
ServletException, IOException {

              response.setContentType("text/html");
             response.setBufferSize(8192);
            PrintWriter out = response.getWriter();
              out.println("<html>" + "<head><title>First Servlet</title></head>");

            out.println("<body>");
            out.println("<h1>Hello J2EE Programmers </H1>");
            out.println("</body>");
            out.close();
            }
      public   void   doPost(HttpServletRequest   request,HttpServletResponse   response)throws
ServletException,IOException{
            doGet(request,response);
}
}
```

Example 2







Web.xml

```
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
        metadata-complete="true">
      <welcome-file-list>
            <welcome-file>test.html</welcome-file>
      </welcome-file-list>
      <servlet>
            <servlet-name>ThreeParams</servlet-name>
            <servlet-class>ThreeParams</servlet-class>
      </servlet>
      <servlet-mapping>
            <servlet-name>ThreeParams</servlet-name>
            <url-pattern>/three</url-pattern>
      </servlet-mapping>
</web-app>
```

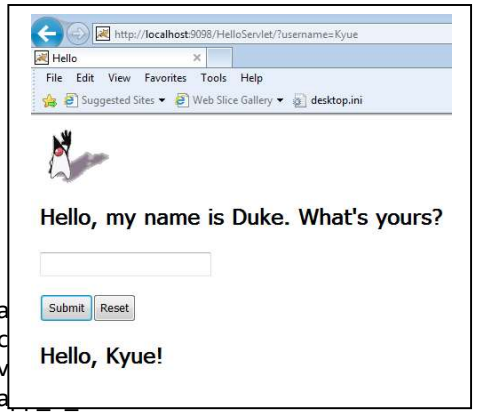
test.html

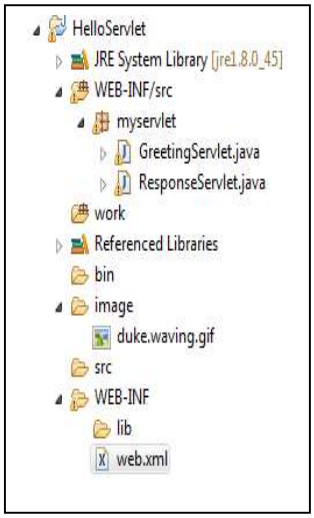
```
<FORM ACTION="http://localhost:9098/Example2/three" METHOD="POST">
First Parameter: <INPUT TYPE="TEXT" NAME="param1"><BR>
Second Parameter: <INPUT TYPE="TEXT" NAME="param2"><BR>
Third Parameter: <INPUT TYPE="TEXT" NAME="param3"><BR>
<CENTER>
<INPUT TYPE="SUBMIT">
</CENTER>
</FORM>
```

ThreeParams.java

```
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.http.*;public class ThreeParams extends HttpServlet{
public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

response.setContentType("text/html");
response.setBufferSize(8192);
PrintWriter out = response.getWriter();
out.println( "<UL>\n" +
"<LI>param1: " + request.getParameter("param1")+ "\n" +
"<LI>param2: " + request.getParameter("param2")+ "\n" +
"<LI>param3: " + request.getParameter("param3")+ "\n" +
"</UL>\n" );
}
public void doPost(HttpServletRequest request,HttpServletResponse response)throws
ServletException,IOException{
doGet(request,response);
}
}
```

Example 3





```
          metadata-complete="true">

        <welcome-file-list>
              <welcome-file>greet</welcome-file>
        </welcome-file-list>
        <servlet>
              <servlet-name>GreetingServlet</servlet-name>
              <servlet-class>myservlet.GreetingServlet</servlet-class>
        </servlet>
        <servlet-mapping>
              <servlet-name>GreetingServlet</servlet-name>
              <url-pattern>/greet</url-pattern>
        </servlet-mapping>
        <servlet>
              <servlet-name>ResponseServlet</servlet-name>
              <servlet-class>myservlet.ResponseServlet</servlet-class>
        </servlet>
        <servlet-mapping>
              <servlet-name>ResponseServlet</servlet-name>
              <url-pattern>/response</url-pattern>
        </servlet-mapping>
</web-app>
```

GreetingServlet.java

```
package myservlet;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class GreetingServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        response.setBufferSize(8192);
        PrintWriter out = response.getWriter();
        out.println("<html>" + "<head><title>Hello</title></head>");
        out.println("<body  bgcolor=\"#ffffff\">" +
            "<img src=\"image/duke.waving.gif\" alt=\"Duke waving\">" +
            "<h2>Hello, my name is Duke. What's yours?</h2>" +
            "<form method=\"get\">" +
            "<input type=\"text\" name=\"username\" size=\"25\">" + "<p></p>" +
            "<input type=\"submit\" value=\"Submit\">" +
            "<input type=\"reset\" value=\"Reset\">" + "</form>");
        String username = request.getParameter("username");
        if ((username != null) && (username.length() > 0)) {
            RequestDispatcher dispatcher=getServletContext().getRequestDispatcher("/response");
           if (dispatcher != null) {
                dispatcher.include(request, response);
            }
       }
out.println("</body></html>");
out.close();
}
public void doPost(HttpServletRequest request,HttpServletResponse response)throws
ServletException,IOException{
doGet(request,response);
}
}
```

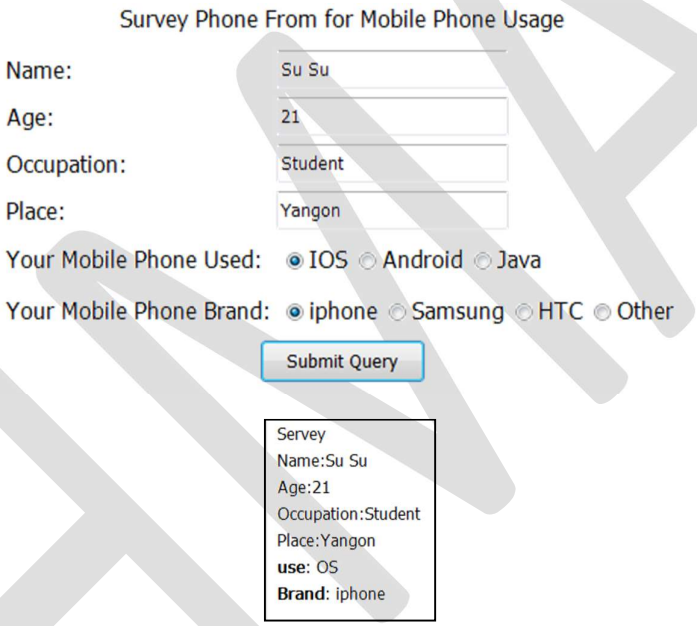
ResponseServlet.java

```
package myservlet;

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

```
public class ResponseServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        PrintWriter out = response.getWriter();

        // then write the data of the response
        String username = request.getParameter("username");

        if ((username != null) && (username.length() > 0)) {
            out.println("<h2>Hello, " + username + "!</h2>");
        }
    }

    public String getServletInfo() {
        return "The Response servlet says hello.";
    }
}
```

Exercise

Exercise 1



Exercise 2

Exercise 3

```
<web-app>
	<welcome-file-list>
		<welcome-file>test.html</welcome-file>
	</welcome-file-list>
	<servlet>
		<servlet-name>ServeyPhone</servlet-name>
		<servlet-class>ServeyPhone</servlet-class>
	</servlet>
	<servlet-mapping>
		<servlet-name>ServeyPhone</servlet-name>
		<url-pattern>/Servey</url-pattern>
	</servlet-mapping>
</web-app>
```

```
<FORM ACTION="http://localhost:9098/UniExercise1/Servey" METHOD="POST">
<table>
	<caption>Survey Phone From for Mobile Phone Usage</caption>
	<tr><td>Name:</td><td> <INPUT TYPE="TEXT" NAME="param1"></td></tr>
	<tr><td>Age: </td><td> <INPUT TYPE="TEXT" NAME="param2"></td></tr>
	<tr><td>Occupation: </td><td> <INPUT TYPE="TEXT" NAME="param3"></td></tr>
	<tr><td>Place:	</td><td> <INPUT TYPE="TEXT" NAME="param4"></td></tr>
	<tr><td>Your Mobile Phone Used:  </td>
		<td>
		<table>
			<tr>
				<td><INPUT TYPE="radio" NAME="r1" value="OS">IOS</td>
				<td><INPUT TYPE="radio" NAME="r1" value="Android">Android</td>
				<td><INPUT TYPE="radio" NAME="r1" value="Java">Java</td>
			</tr>
		</table>

		</td>
	</tr>
	<tr><td>Your Mobile Phone Brand: </td>
		<td>
		<table>
			<tr>
				<td><INPUT TYPE="radio" NAME="r2" value="iphone">iphone</td>
				<td><INPUT TYPE="radio" NAME="r2" value="Samsung">Samsung</td>
				<td><INPUT TYPE="radio" NAME="r2" value="HTC">HTC</td>
				<td><INPUT TYPE="radio" NAME="r2" value="Other">Other</td>
			</tr>
		</table>
		</td>
	</tr>
	<tr align="center"><td colspan="2"><INPUT TYPE="SUBMIT"></td></tr>
</table>
</FORM>
```

```
import java.io.*;
import java.util.Enumeration;

import javax.servlet.ServletException;
import javax.servlet.http.*;
public class ServeyPhone  extends HttpServlet{
	public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
		// Just send back a simple HTTP response
		 response.setContentType("text/html");
		 response.setBufferSize(8192);
		PrintWriter out = response.getWriter();
		String name= request.getParameter("param1");
		String age=request.getParameter("param2");
```

```
            String occupation= request.getParameter("param3");
            String place=request.getParameter("param4");


            out.println( "Servey<br>");
            out.println( "Name:" +  name+"<br>");
            out.println( "Age:" +  age+"<br>");
            out.println( "Occupation:" +  occupation+"<br>");
            out.println( "Place:" +  place+"<br>");


            String os[]=request.getParameterValues("r1");
            out.println("<b>use</b>:");
            for(int i=0;i<os.length;i++)
                  out.println(os[i]);

            String brand[]=request.getParameterValues("r2");
            out.println("<br><b>Brand</b>:");
            for(int i=0;i<brand.length;i++)
                  out.println(brand[i]);
            out.close();
      }
public void doPost(HttpServletRequest request,HttpServletResponse response)throws
ServletException,IOException{
      doGet(request,response);
}
}
```

Exercise 4

```
<web-app>
      <welcome-file-list>
            <welcome-file>test.html</welcome-file>
      </welcome-file-list>
      <servlet>
            <servlet-name>ServeyPhone</servlet-name>
            <servlet-class>ServeyPhone</servlet-class>
      </servlet>
      <servlet-mapping>
            <servlet-name>ServeyPhone</servlet-name>
            <url-pattern>/Servey</url-pattern>
      </servlet-mapping>
</web-app>

<FORM ACTION="http://localhost:9098/UniExercise2/Servey" METHOD="POST">
<table>
      <caption>Student Registration</caption>
      <tr><td>Name:   </td><td> <INPUT TYPE="TEXT" NAME="param1"></td></tr>
      <tr><td>Age:    </td><td> <INPUT TYPE="TEXT" NAME="param2"></td></tr>
      <tr><td>Occupation:     </td><td> <INPUT TYPE="TEXT" NAME="param3"></td></tr>
      <tr>
            <td>Address:     </td>
            <td>
                  <select name="place">
                        <option value="Mandalay">Mandalay</option>
                        <option value="Yangon" selected>Yangon</option>
                        <option value="Nay Pyay Taw">Nay Pyay Taw</option>
                        <option value="Ayeyarwady">Ayeyarwady</option>
                  </select>
            </td>
      </tr>
      <tr><td>Gender: </td>
            <td>
            <table>
                  <tr>
                        <td><INPUT TYPE="radio" NAME="r1" value="male">male</td>
                        <td><INPUT TYPE="radio" NAME="r1" value="female">female</td>
```

```
                    </tr>
                </table>

                </td>
            </tr>
            <tr><td>Your Hobby:      </td>
                <td>
                <table>
                    <tr>
                        <td><input type="checkbox" name="hobby" value="Driving" checked> Driving</td>
                        <td><input type="checkbox" name="hobby" value="Reading"> Reading</td>
                        <td><input type="checkbox" name="hobby" value="Singing"> Singing</td>
                        <td><input type="checkbox" name="hobby" value="Playing Games"> Playing Games</td>
                    </tr>
                </table>
                </td>
            </tr>
            <tr align="center"><td colspan="2"><INPUT TYPE="SUBMIT"></td></tr>
</table>
</FORM>

import java.io.*;
import java.util.Enumeration;
import javax.servlet.ServletException;
import javax.servlet.http.*;
public class ServeyPhone  extends HttpServlet{
        public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
                // Just send back a simple HTTP response
                 response.setContentType("text/html");
            response.setBufferSize(8192);
                PrintWriter out = response.getWriter();
                String name= request.getParameter("param1");
                String age=request.getParameter("param2");
                String occupation= request.getParameter("param3");


                out.println( "Student Information<br>");
                out.println( "Name:" +  name+"<br>");
                out.println( "Age:" +  age+"<br>");
                out.println( "Occupation:" +  occupation+"<br>");

                String place=request.getParameter("place");
                out.println("Address:"+place+"<br>");

                String gender[]=request.getParameterValues("r1");
                out.println("<b>Gender</b>:");
                for(int i=0;i<gender.length;i++)
                        out.println(gender[i]);

                String hobby[]=request.getParameterValues("hobby");
                out.println("<br><b>Hobby</b>:");
                for(int i=0;i<hobby.length;i++)
                        out.println(hobby[i]);
                out.close();
        }
public void doPost(HttpServletRequest request,HttpServletResponse response)throws ServletException,IOException{
        doGet(request,response);
}
}
```