Session 5

Introduction to Servlets

1

Lecture Objectives

- Understand the popular approaches to generating HTML on a server
- Know how the Hello World servlet operates
- Understand the interaction among the browser, Web server, application server, and servlet code
- Understand the servlet life cycle

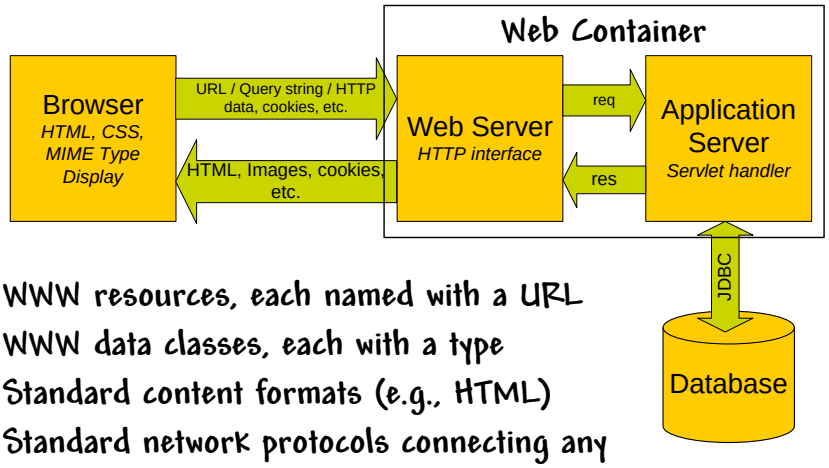
2

Reading & Reference

- Head First Servlets & JSP
  - Chapter 1 & 2 J2EE 6
- Reference
  - Use the on-line Servlet API documentation at: http://docs.oracle.com/javaee/6/api/
  - http spec www.ietf.org/rfc/rfc2616.txt?number=2616

3

Typical Current Web Architecture



- WWW resources, each named with a URL
- WWW data classes, each with a type
- Standard content formats (e.g., HTML)
- Standard network protocols connecting any browser with any server

4

Why Do We Need To Generate HTML?

- Include information from databases and mainframe systems (shopping sites)
- Include information from Web services
- Generate personalized content (e.g. myYahoo and Amazon)
- Generate content common to multiple pages

5

Strategies to Generate HTML

- Common Gateway Interface (CGI)
  - HTML request triggers the execution of a script
  - Old technology
  - New process for every request
  - Limited access to server data
- Server scripting
  - Microsoft ASP.NET
  - Java Servlet / JSP

The concepts used in the Java and Microsoft environments are very similar

6

What is a Servlet?

- A Java class that can be loaded dynamically to expand the capability of the Web server
- Runs inside the Java Virtual Machine on the server (safe and portable)
- Able to access all Java APIs supported in the server
- Does not have a main method (just like an applet)

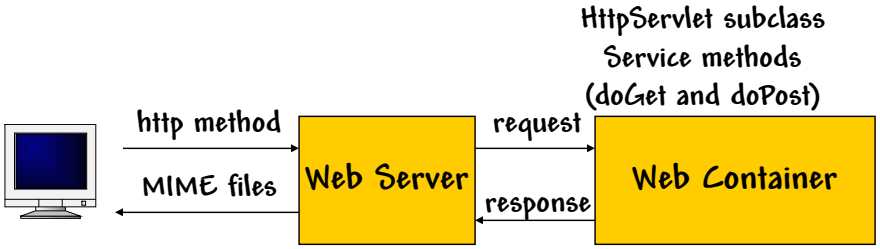
7

Servlet Implementation

- Some Platforms (Web Containers)
  - IBM WebSphere
  - BEA WebLogic
  - Apache (Jakarta TomCat)
  - Glassfish
  - JBoss

8

Servlet / Web Server Interface



The Web Server primarily serves resources, and passes complex request to the Application Server

Sometimes both servers are packaged together, but it is good to think of them as separate systems

9

Servlet Interface

- Objects are used to pass information to the server and to return information from the server
- Service methods:

```
protected void doGet(
    HttpServletRequest req,
    HttpServletResponse resp)
    throws ServletException,
           java.io.IOException

protected void doPost(
    HttpServletRequest req,
    HttpServletResponse resp)
    throws ServletException,
           java.io.IOException
```

Request data includes HTTP version, URL, browser software, client MIME type preferences, data, etc.

Response data includes HTTP version, status code, MIME type of data, document size, document

10

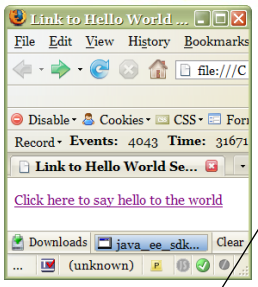
Example as an HML page

- The following HTML can be returned to the browser directly by the Web server (static file on the Web Server) - or the same html page can be generated (on the fly) by the Web Container

```
<html>
<head>
    <title>Hello WWW</title>
</head>
<body>
    <p>Hello WWW</p>
</body></html>
```


11

Invoking the Hello World Servlet



This URL in this link maps to the servlet

Verify the port number used by your test server

Name of the Web application

```
<?xml version="1.0" encoding="iso-8859-1"?> <!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head> <title>Link to Hello World Servlet</title> </head>
<body> <p>
<a href="http://localhost:8080/CodeCSE336/helloWWW.html">
Click here to say hello to the world</a> </p> </body>
</html>
```

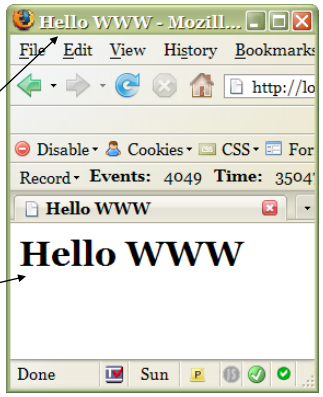
This is not really a file - it maps to your servlet

12

Hello WWW Servlet Method

```
protected void processRequest(
  HttpServletRequest request,
  HttpServletResponse response)
  throws ServletException, IOException {

  response.setContentType(
    "text/html;charset=UTF-8");
  PrintWriter out = response.getWriter();
  String docType =
    "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD"
   + "HTML 4.0 Transitional//EN\">\n";
  out.println(docType);
  out.println("<html>");
  out.println("<head><title>"
    + "Hello WWW</title></head>");
  out.println("<body>");
  out.println("<h1>Hello WWW</h1>");
  out.println("</body></html>");
  out.close();
  }
```



13

Hello WWW Servlet Class

```
package lectures;
import java.io.*;import java.net.*;import javax.servlet.*;
import javax.servlet.http.*;

public class HelloWWW extends HttpServlet {
protected void processRequest(HttpServletRequest request,
   HttpServletResponse response)
  throws ServletException, IOException {
}
protected void doGet(HttpServletRequest request,
   HttpServletResponse response)
  throws ServletException, IOException {
    processRequest(request, response);
  }
protected void doPost(HttpServletRequest request,
   HttpServletResponse response)
  throws ServletException, IOException {
    processRequest(request, response);
  }
```

processRequest is a method used by convention in NetBeans

The Web container calls either doGet or doPost, which then calls processRequest

14

Web Application

- Your Web application is stored in a directory (and deployed as a war file)
- Top level directory of the Web application is the document root of the application, containing JSP pages and static Web resources (or subdirectories of JSP, etc.)
- Document root contains a sub-directory called WEB-INF, containing
  - web.xml - the deployment descriptor
  - classes - a directory containing server classes (e.g., servlets)
  - lib - a directory that contains JAR archives of libraries
- Package directories can be either in the document root or the WEB-INF/classes directory

Take a look at your Web App in your NetBeans or Eclipse project pane

15

How to Specify the Servlet in Your HTML

- A URL is used to request that the container run your servlet (in an anchor tag or form tag)
- URL contains the host name, port (optional), and path
- In a servlet container, the path can be mapped (what you see is not always what you get)

`http://localhost:8080/CodeCSE336/helloWWW.html`

There is no helloWWW file

16

How URLs Run Servlets

http://localhost:8080/CodeCSE336/helloWWW.html

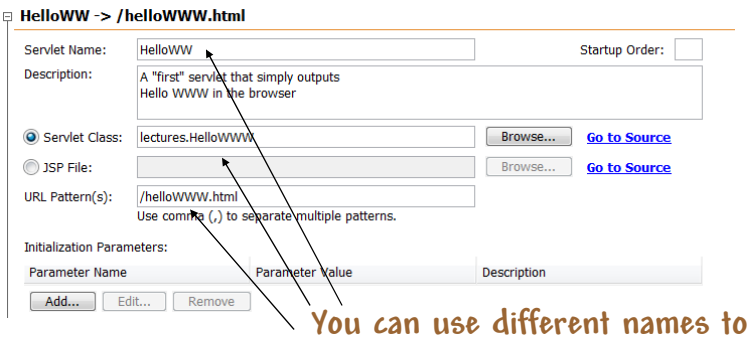
Context name

- The servlet container evaluates the URL request to see if the first part of the path matches a context name
- If the path matches a context name, the context name is mapped to a Web application root directory (using the web.xml deployment descriptor)

17

Deployment Descriptor (web.xml)

- Used to deploy your Web application (i.e., servlets, JSPs, etc.)
- NetBeans display of Deployment Descriptor (below)



You can use different names to identify the servlet in different places

18

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
   http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
 <servlet>
   <description>A "first" servlet that simply outputs
        Hello WWW in the browser</description>
   <servlet-name>HelloWorld</servlet-name>
   <servlet-class>lectures.HelloWWW</servlet-class>
 </servlet>
 <servlet-mapping>
   <servlet-name>HelloWWW</servlet-name>
   <url-pattern>/helloWWW.html</url-pattern>
 </servlet-mapping>
 <session-config>
   <session-timeout>30</session-timeout>
 </session-config>
 <welcome-file-list>
   <welcome-file>index.jsp</welcome-file>
 </welcome-file-list>
</web-app>
```

19

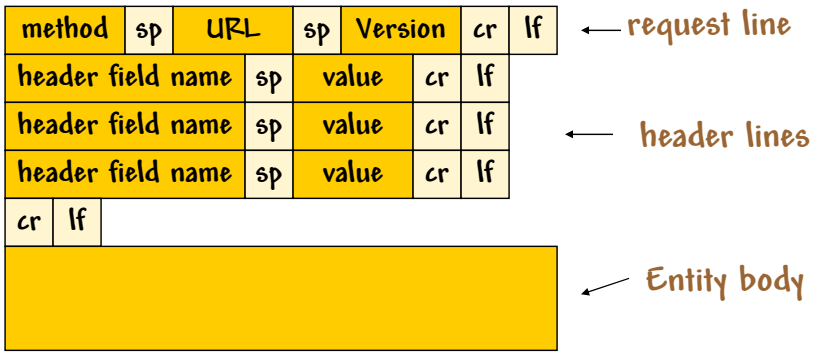
Servlet Life Cycle

- Initialization - by the Web server or the first invocation of the servlet
  - init method is executed when servlet is started
  - Servlet object remains alive to handle requests
- service method - called by the Web Server
  - Checks the http method and calls doGet, doPost, etc.
  - Also handles HEAD, OPTIONS, and TRACE requests
- Termination - the destroy method is called by the Web server prior to termination of the servlet

21

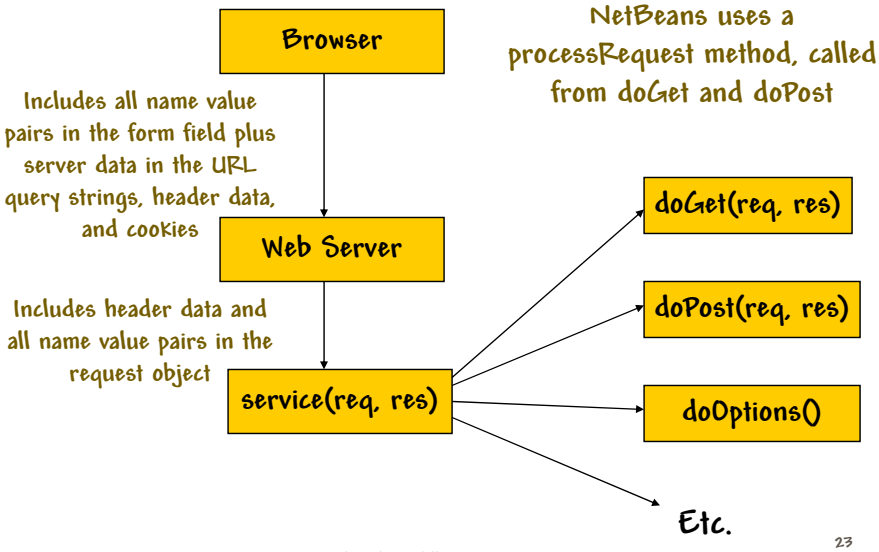
Request Message Format

- The http request is specified by the request line, a variable number of header fields, and the entity body



22

Calling Sequence



23

Some HttpServletRequest Methods

ServletRequest

- getParameter(String s)
- getScheme()
- getProtocol()
- getRemoteAddress()
- isSecure()
- getContentType()

HttpServletRequest

- getRequestedSessionID()
- getRequestURI
- isRequestedSessionValid()
- getQueryString()
- getRemoteUser
- getMethod()
- getCookies()
- getHeader()

24

HttpServletResponse

- Some methods:
  - getWriter - from ServletResponse
  - sendError(int sc)
  - addCookie(Cookie cookie)
  - sendRedirect(String location)
- Some fields:
  - SC_GONE
  - SC_INTERNAL_SERVER_ERROR
  - SC_NOT_FOUND

25

Why doGet and doPost?

- HTTP - a simple stateless protocol (Web browser makes a request and the server responds)
- The request from the browser specifies an HTTP method, along with client data
- HTTP methods - GET, POST, HEAD, etc.
- Method called (doGet or doPost) corresponds to the HTTP method requested by the browser

26

doGet / doPost Practice

- A servlet usually does not distinguish between a GET and a POST method call
- One of the methods usually invokes the other

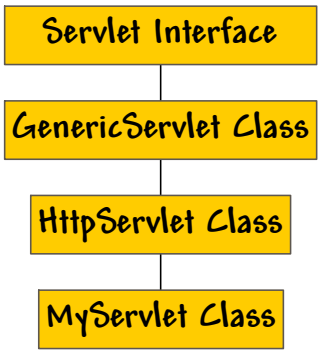
```
public void doPost(HttpServletRequest request,
      HttpServletResponse response)
      throws ServletException, IOException {
   this.doGet(request, response);
 }
```

Or NetBeans generates a processRequest method

```
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
  throws ServletException, IOException {
    processRequest(request, response);
  }
```


27

Servlet Hierarchy



28

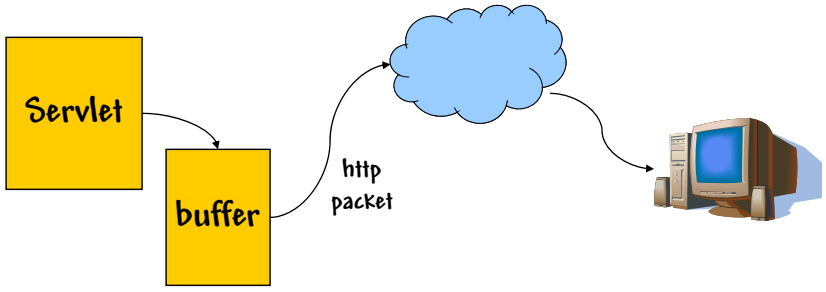
Servlet Interface

- doGet, doPost
- init(ServletConfig) - initializes the servlet
- service(ServletRequest, ServletResponse)
- getServletConfig() - returns the servlet's configuration object
- getServletInfo() - returns information about the servlet (used primarily for system administration)
- destroy() - cleans up resources held by the servlet

29

Servlet Generation of HTML

- A servlet will generate 2 kinds of output
  - Information about the transmission to the browser - this is stored in the http header of the response
  - Data (usually HTML) that is stored in the http body of the response



30

Server Stream Caching

- Header data - headers can be set in any order and are not sent until the first buffer fills
  - Response header data includes age, cache control, language, message digest, MIME type, expiration date, last modified date, etc.
  - isCommitted method - returns boolean indicating that headers have been sent
- Buffered stream data
  - setBufferSize
  - flushBuffer

31

Have You Satisfied the Lecture Objectives?

- Understand the popular approaches to generating HTML on a server
- Know how the Hello World servlet operates
- Understand the interaction among the browser, Web server, application server, and servlet code
- Understand the servlet life cycle

32