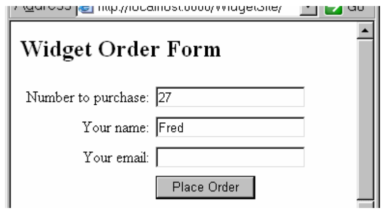
Server-side Web Development and Programming

Lecture 5:
Java Servlets and the Control-View Architecture

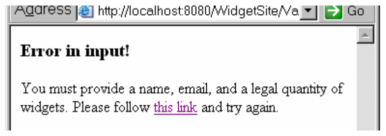
The Control-View Architecture

- Different user input might require <u>different</u> response pages
  - Different types of request
  - Errors/missing field values, etc.
    - Example: missing fields in Widget order


http://localhost:8080/WidgetSite/

Widget Order Form

Number to purchase:	27
Your name:	Fred
Your email:	


http://localhost:8080/WidgetSite/Va

Error in input!

You must provide a name, email, and a legal quantity of widgets. Please follow this link and try again.

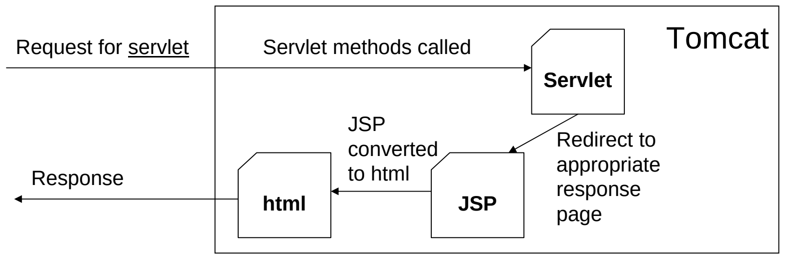
The Control-View Architecture

- Bad solution: single JSP with lots of conditions

```
<% if (fields are valid) { %>
   entire web page for normal response
<% } else { %>
   entire web page for error message(s)
<% } %>
```

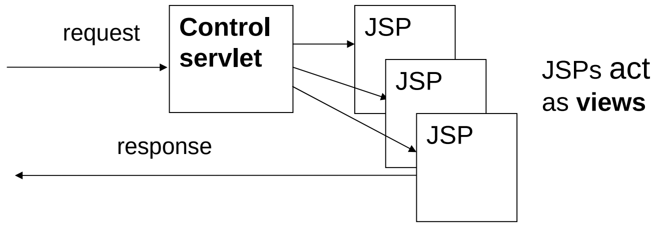
Java Servlets

- <u>Class</u> containing methods executed by Tomcat
  - Not a web page (like a JSP)
  - Methods invoked by a request for the servlet
  - Usually <u>redirects</u> to a JSP



The Control-View Architecture

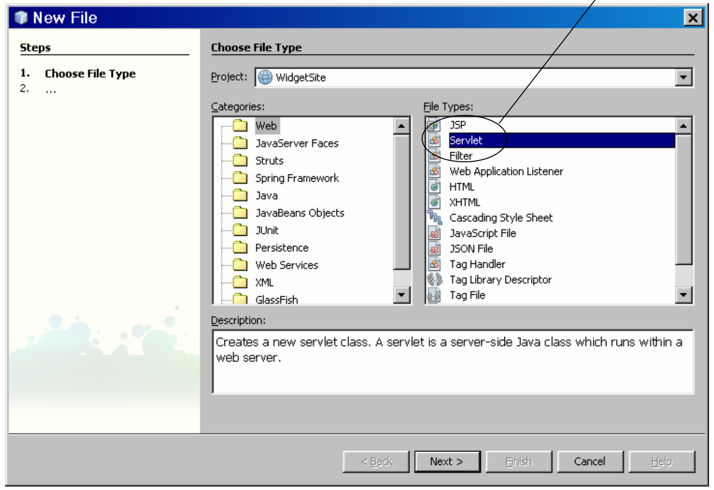
- Servlets usually act as <u>controls</u>
  - Categorize request based on the parameters and possibly other factors (database info, etc.).
  - Decide which JSP should be sent back as response.
  - Forward control (and request data) to that JSP.



Adding a Servlet

- File → New File

Choose Servlet type


New File

Steps
1. Choose File Type
2. ...

Choose File Type

Project: WidgetSite

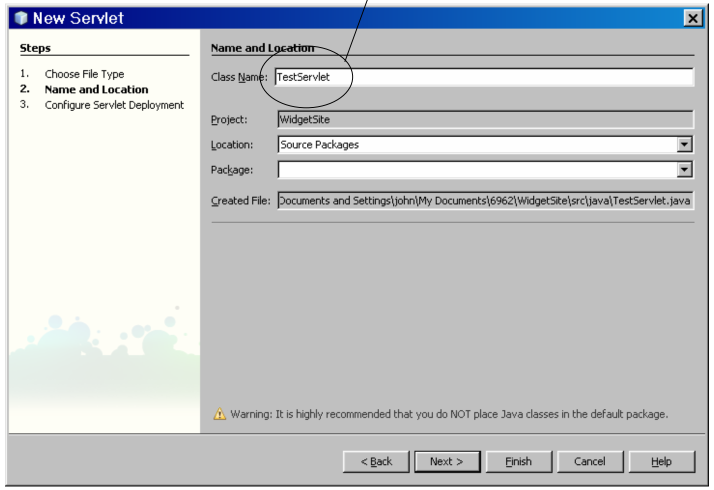
Categories: Web, JavaServer Faces, Struts, Spring Framework, Java, JavaBeans Objects, JUnit, Persistence, Web Services, XML, GlassFish

File Types: JSP, Servlet, Filter, Web Application Listener, HTML, XHTML, Cascading Style Sheet, JavaScript File, JSON File, Tag Handler, Tag Library Descriptor, Tag File

Description:
Creates a new servlet class. A servlet is a server-side Java class which runs within a web server.

Adding a Servlet

Give it a name


New Servlet

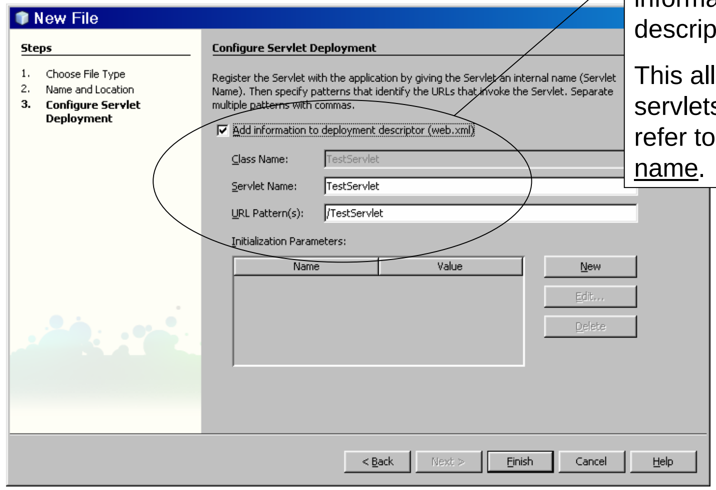
Steps
1. Choose File Type
2. **Name and Location**
3. Configure Servlet Deployment

Name and Location

Class Name:	TestServlet
Project:	WidgetSite
Location:	Source Packages
Package:	
Created File:	Documents and Settings\john\My Documents\6962\WidgetSite\src\java\TestServlet.java

Warning: It is highly recommended that you do NOT place Java classes in the default package.

Adding a Servlet


New File

Steps
1. Choose File Type
2. Name and Location
3. **Configure Servlet Deployment**

Configure Servlet Deployment

Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.

☑ Add information to deployment descriptor (web.xml)

Class Name:	TestServlet
Servlet Name:	TestServlet
URL Pattern(s):	/TestServlet

Initialization Parameters:

Name	Value

Adds servlet information to <u>web.xml</u> descriptor file.

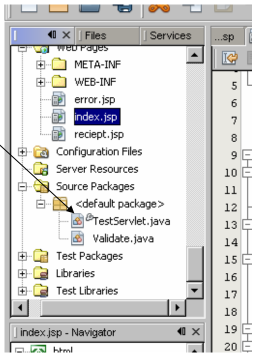
This allows other servlets and JSPs to refer to it using its <u>name</u>.

Adding a Servlet

- Servlet added to <u>source packages</u> of site
- When deployed, must be in <u>WEB-INF/classes</u> subdirectory of site

webapps→
  application directory →
    your html files and Java Server pages
    WEB-INF →
      web.xml
      classes →
        ***yourservlet*.class**

  – **Note that the *yourservlet.java* file must be <u>compiled</u> to create *yourservlet.class***



Basic Servlet Structure

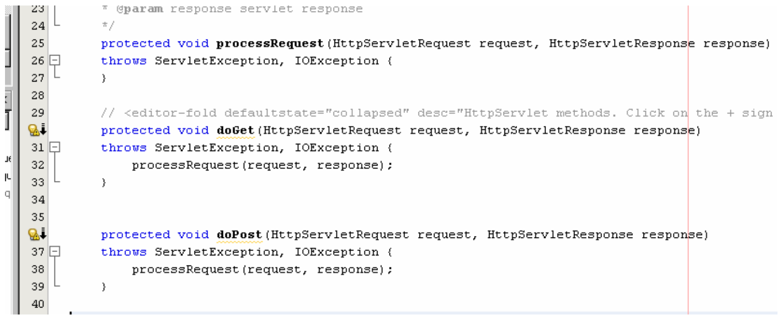
Key methods:

- `void doGet(HttpServletRequest request,`
  `HttpServletResponse response)`
  Called if servlet invoked using <u>get</u> method
- `void doPost(HttpServletRequest request,`
  `HttpServletResponse response)`
  Called if servlet invoked using <u>post</u> method

- Have access to <u>request</u> object
  - Can call `getParameter`, etc.

Basic Servlet Structure

- Note that 99.9% both `doGet` and `doPost` do <u>same thing</u>
- NetBeans generates code in both that just calls single <u>processRequest</u> method.
  - `doGet` and `doPost` hidden by editor



```java
     * @param response servlet response
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign
    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);
    }

```

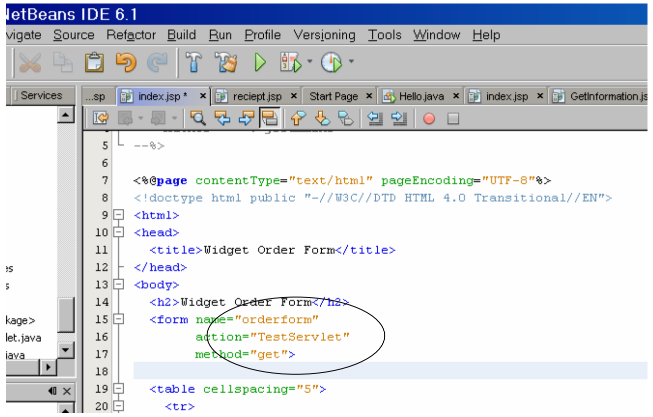
Importing Servlet Libraries

- Servlets libraries generally imported:
  - `import java.io.*;`
  - `import javax.servlet.*;`
  - `import javax.servlet.http.*;`
    This is where request, response, etc. defined

- Note that NetBeans does not automatically import these (just specific classes)
  - Should change code to include <u>all</u> of these

Invoking a Servlet from a JSP

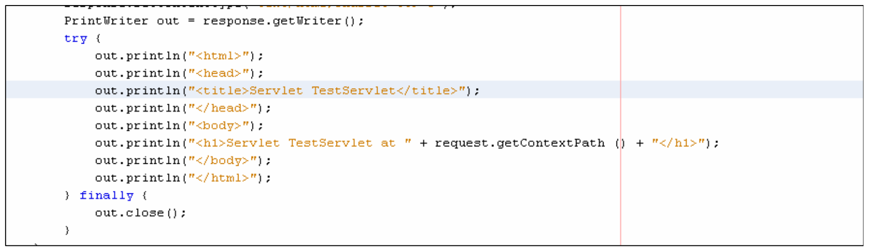
- Use its <u>name</u> in the ACTION attribute of FORM



```
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
  <title>Widget Order Form</title>
</head>
<body>
  <h2>Widget Order Form</h2>
  <form name="orderform"
        action="TestServlet"
        method="get">

  <table cellspacing="5">
    <tr>
```

Servlet Background

- Preceded development of JSP model
  - Modeled after CGI-BIN model
- Can generate <u>own</u> response page by writing a string of html to <u>response</u> object



```
PrintWriter out = response.getWriter();
try {
    out.println("<html>");
    out.println("<head>");
    out.println("<title>Servlet TestServlet</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Servlet TestServlet at " + request.getContextPath () + "</h1>");
    out.println("</body>");
    out.println("</html>");
} finally {
    out.close();
}
```

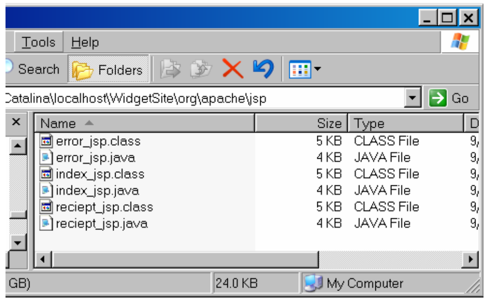
  - Very rarely done!
  - Usually just redirect to JSP to create response

Servlet Background

- JSP model built on servlets
  - When JSP called for <u>first time</u>
    - JSP converted to equivalent servlet and compiled
    - Stored in <u>WORK</u> directory
    - <u>Run</u> to generate html for response

Only this done in subsequent requests

Much more efficient than running JSP again each request



Catalina\localhost\WidgetSite\org\apache\jsp

Name	Size	Type	D
error_jsp.class	5 KB	CLASS File	9,
error_jsp.java	4 KB	JAVA File	9,
index_jsp.class	5 KB	CLASS File	9,
index_jsp.java	4 KB	JAVA File	9,
reciept_jsp.class	5 KB	CLASS File	9,
reciept_jsp.java	4 KB	JAVA File	9,

Servlet Redirection

Basic syntax (step 1):

```
RequestDispatcher dispatcherObject =
 getServletContext().
       getRequestDispatcher("/MyJSP");
```

Forward control to this JSP on the same site

The / tells Tomcat to look in the application root directory

dispatcherObject is a new object that holds information about the redirection

Get the location of the site (so can do a relative url forward)

Servlet Redirection

Basic syntax (step 2):

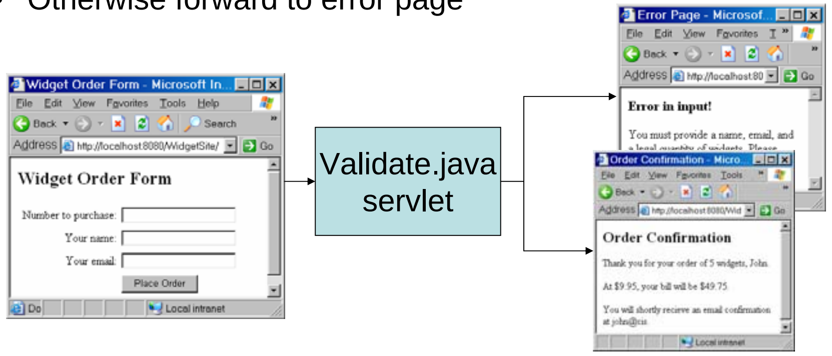
```
dispatcherObject.forward(request, response);
```

Transfer control using the *dispatcherObject*

Both the request and response must be passed so the JSP has access to parameters, etc.

Redirection Example

- `index.jsp` prompts for quantity, name, email
- Upon submit, invokes `Validate.java` servlet
- If all information present, forward to receipt page
- Otherwise forward to error page


Widget Order Form - Microsoft In...
File Edit View Favorites Tools Help
Back Search
Address http://localhost:8080/WidgetSite/ Go
Validate.java servlet
Error Page - Microsof...
Address http://localhost:80 Go
Order Confirmation - Micro...
Address http://localhost:8080/Wid Go
Local intranet

Widget Order Form

Number to purchase:

Your name:

Your email:

Place Order

Error in input!

You must provide a name, email, and a legal quantity of widgets. Please

Order Confirmation

Thank you for your order of 5 widgets, John.

At $9.95, your bill will be $49.75

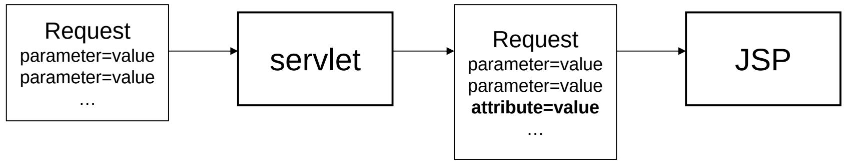
You will shortly recieve an email confirmation at john@cis

Redirection Example

```
public class Validate extends HttpServlet {
    protected void processRequest(HttpServletRequest request,
                                  HttpServletResponse response)
                           throws ServletException, IOException {
      String url = "";  // url to forward to

      // Get the parameter values from the request
      String name = request.getParameter("customerName");
      String email = request.getParameter("customerEmail");
      String quantity = request.getParameter("quantity");

      // If any are empty, set the url to forward to to the error page.
      // Otherwise, forward to the normal reciept
      if (name.equals("") || email.equals("") || quantity.equals("")) {
          url = "/error.jsp";
      }
      else {url = "/reciept.jsp";}

      // Create the dispatcher from the url and perform the forward
      RequestDispatcher dispatcher =
                    getServletContext().getRequestDispatcher(url);
      dispatcher.forward(request, response);
    }
```

Passing Information to the JSP

- Information can be passed from a <u>servlet</u> to the JSP it forwards to
- Added to <u>request</u> object as an <u>attribute</u>
  - Like a parameter, has <u>name</u> and a <u>value</u>
  - Value can be <u>any Java object</u> (not just a string)

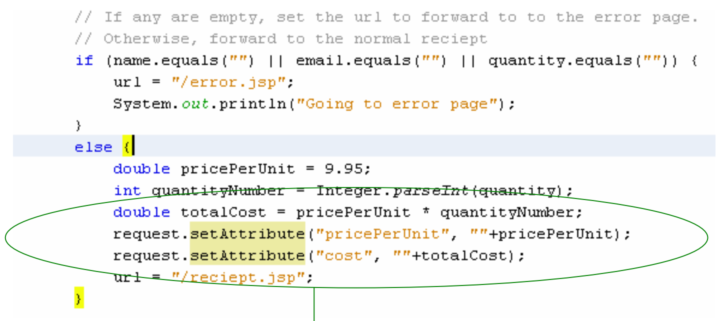


Passing Information to the JSP

- Adding attribute in servlet:
  `request.setAttribute("name", some object);`
- Retrieving attribute in JSP:
  `variable = (type)request.getAttribute("name");`

Since attribute can be <u>any</u> type, must use casting to tell Java original type

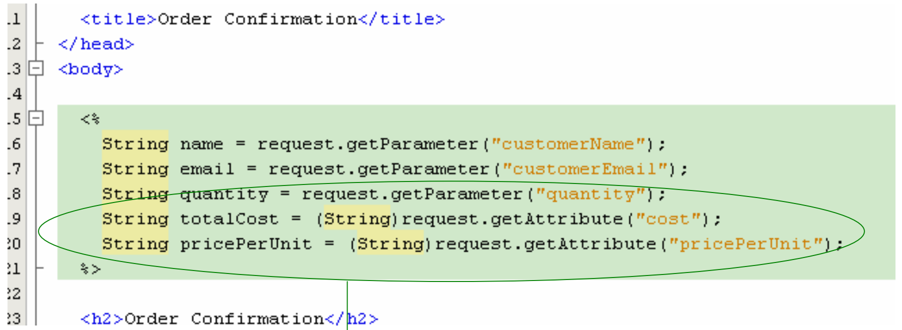
Passing Information to the JSP



```java
// If any are empty, set the url to forward to to the error page.
// Otherwise, forward to the normal reciept
if (name.equals("") || email.equals("") || quantity.equals("")) {
    url = "/error.jsp";
    System.out.println("Going to error page");
}
else {
    double pricePerUnit = 9.95;
    int quantityNumber = Integer.parseInt(quantity);
    double totalCost = pricePerUnit * quantityNumber;
    request.setAttribute("pricePerUnit", ""+pricePerUnit);
    request.setAttribute("cost", ""+totalCost);
    url = "/reciept.jsp";
}
```

Code in servlet to pass price per unit and total cost as strings

Passing Information to the JSP



```jsp
    <title>Order Confirmation</title>
  </head>
  <body>

    <%
      String name = request.getParameter("customerName");
      String email = request.getParameter("customerEmail");
      String quantity = request.getParameter("quantity");
      String totalCost = (String)request.getAttribute("cost");
      String pricePerUnit = (String)request.getAttribute("pricePerUnit");
    %>

    <h2>Order Confirmation</h2>
```

Code in JSP to retrieve price per unit and total cost as strings

Servlet Details

- Important note: Forward does <u>not</u> terminate servlet!
  - Will run to end of `processRequest` even after forward
- Bad code:

```
if (somevalue == null) {
    forward to error page
}
code that will crash if somevalue is null
```

- Better code:

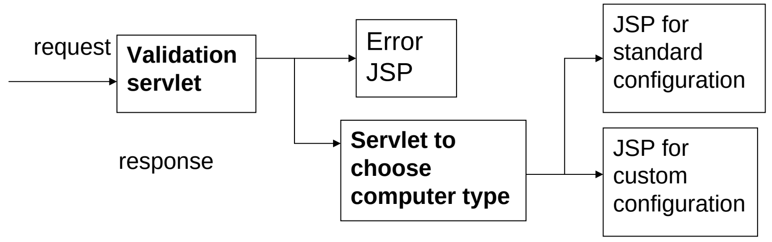
```
if (somevalue == null) {
    forward to error page
}
else {
    code that will crash if somevalue is null
}
```

Servlet Details

- Invoking one servlet from another:
  - `getRequestDispatcher("/sitename/servletname");`

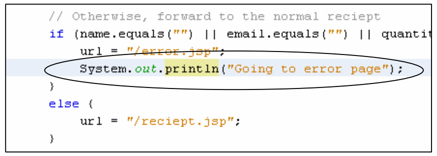
    WidgetSite — Validate

  - Note: may not need sitename in NetBeans, but may not work when deployed otherwise
  - Often done for modular multistage redirection

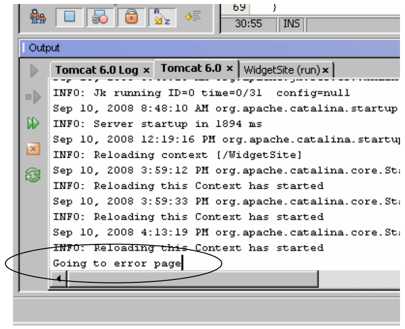

request
Validation servlet
response
Error JSP
Servlet to choose computer type
JSP for standard configuration
JSP for custom configuration

Servlet Details

- Debugging servlets
  - Can write diagnostic messages to control screen
  - `System.out.println("message");`



```
// Otherwise, forward to the normal reciept
if (name.equals("") || email.equals("") || quantit
    url = "/error.jsp";
    System.out.println("Going to error page");
}
else {
    url = "/reciept.jsp";
}
```



```
INFO: Jk running ID=0 time=0/31  config=null
Sep 10, 2008 8:48:10 AM org.apache.catalina.startup
INFO: Server startup in 1894 ms
Sep 10, 2008 12:19:16 PM org.apache.catalina.startu
INFO: Reloading context [/WidgetSite]
Sep 10, 2008 3:59:12 PM org.apache.catalina.core.St
INFO: Reloading this Context has started
Sep 10, 2008 3:59:33 PM org.apache.catalina.core.St
INFO: Reloading this Context has started
Sep 10, 2008 4:13:19 PM org.apache.catalina.core.St
INFO: Reloading this Context has started
Going to error page
```