

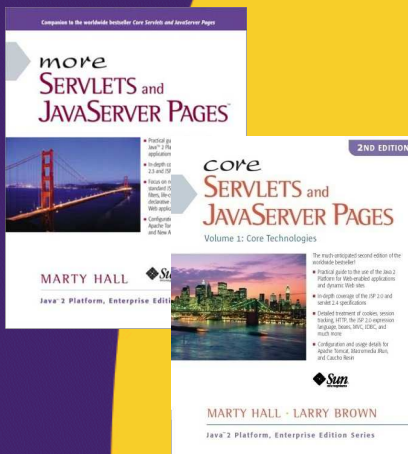


Servlet and JSP Review

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



For live Ajax training, please see training courses at <http://courses.coreservlets.com/>.



Taught by the author of *Core Servlets and JSP*, *More Servlets and JSP*, and this tutorial. Available at public venues, or customized versions can be held on-site at your organization.

- Courses developed and taught by Marty Hall
 - Java 5, Java 6, intermediate/beginning servlets/JSP, advanced servlets/JSP, Struts, JSF, Ajax, GWT, custom courses.
- Courses developed and taught by coreservlets.com experts (edited by Marty)
 - Spring, Hibernate, EJB3, Ruby/Rails

Contact hall@coreservlets.com for details

Agenda

- What servlets are all about
- Servlet basics
- Creating and deploying projects
- Creating forms and reading form data
- JSP scripting
- Using XML syntax for JSP pages
- JSP file inclusion
- MVC

4

Java EE training: <http://courses.coreservlets.com>

© 2008 Marty Hall



Servlet Basics

Customized Java EE Training: <http://courses.coreservlets.com/>

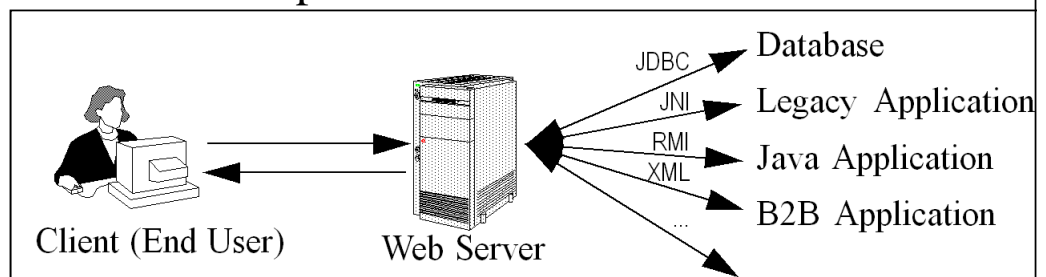
Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

5

A Servlet's Job

- **Read explicit data sent by client**
 - Form data
- **Read implicit data sent by client**
 - Request headers
- **Generate the results**
- **Send the explicit data back to client**
 - HTML or XML or JSON or custom data format
- **Send the implicit data to client**
 - Status codes and response headers



6

Accessing the Online Documentation

- **Servlets and JSP**
 - http://java.sun.com/products/servlet/2.5/docs/servlet-2_5-mr2/
 - http://java.sun.com/products/jsp/2.1/docs/jsp-2_1-pfd2/
 - <http://tomcat.apache.org/tomcat-5.5-doc/servletapi/>
 - <http://tomcat.apache.org/tomcat-5.5-doc/jspapi/>
- **Java 6 or Java 5**
 - <http://java.sun.com/javase/6/docs/api/>
 - Class uses Java 6 and Tomcat 6
 - <http://java.sun.com/j2se/1.5.0/docs/api/>
- **Advice**
 - If you have a fast and reliable internet connection, bookmark these addresses
 - If not, download a copy of the APIs onto your local machine and use it

7

A Sample Servlet (Code)

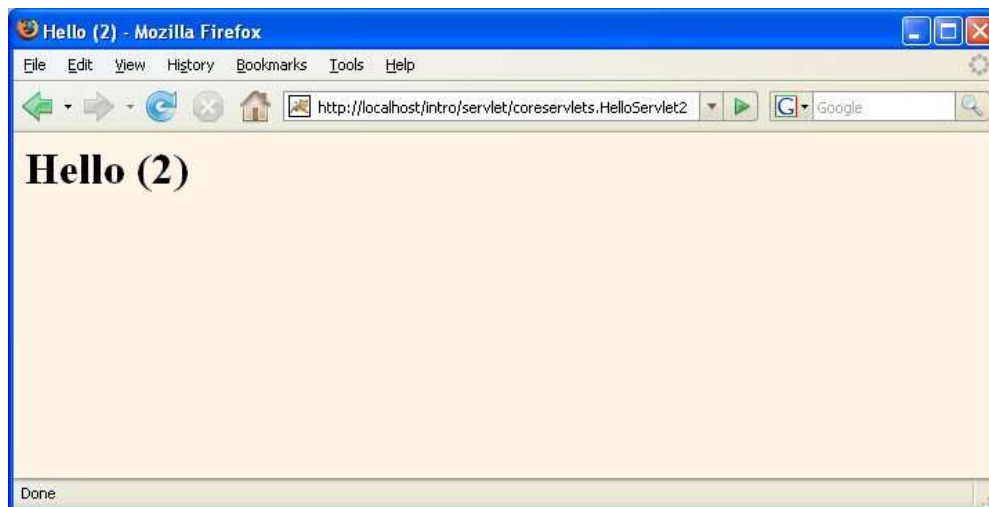
```
package coreservlets;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloServlet2 extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String docType =
            "<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 "+
            "Transitional//EN">\n";
        out.println(docType +
            "<HTML>\n" +
            "<HEAD><TITLE>Hello (2)</TITLE></HEAD>\n"+
            "<BODY BGCOLOR=\"#FDF5E6\">\n" +
            "<H1>Hello (2)</H1>\n" +
            "</BODY></HTML>");
    }
}
```

8

Java EE training: <http://courses.coreservlets.com>

A Sample Servlet (Result)



Assumes Eclipse project named intro. Code in src/coreservlets/HelloServlet2.java.

If you make the web.xml entries from the upcoming slides, you could also use the URL <http://localhost/intro/hi2>

9

Java EE training: <http://courses.coreservlets.com>



Testing with Eclipse

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

10

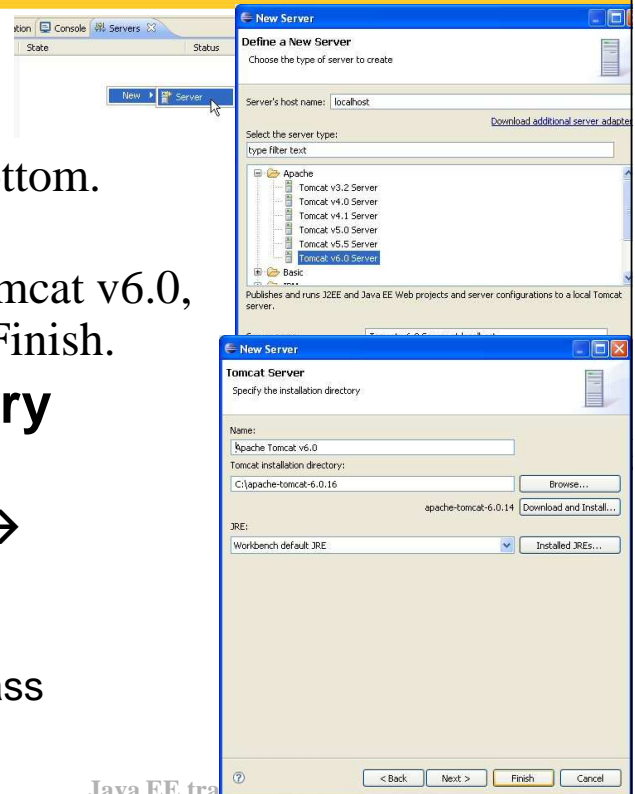
Installing Eclipse

- **Overview**
 - Eclipse is a free open-source development environment with support for Java and many other languages
- **Downloading**
 - <http://www.eclipse.org/downloads/>
 - Choose "Eclipse IDE for Java EE Developers"
 - As of 8/2008, version 3.4, called Eclipse Ganymede
- **Installing**
 - Unzip into directory of your choice
 - Put shortcut to eclipse.exe on your desktop
- **Integrating Tomcat in Eclipse**
 - <http://www.coreservlets.com/Apache-Tomcat-Tutorial/eclipse.html>

11

Configuring Eclipse

- **Make sure Eclipse knows about Tomcat**
 - Click on Servers tab at bottom. R-click in window.
 - New, Server, Apache, Tomcat v6.0, Next, navigate to folder, Finish.
- **Suppress unnecessary compiler warnings**
 - Window → Preferences → Java → Compiler → Errors/Warnings
 - Change "Serializable class without ..." to "Ignore"

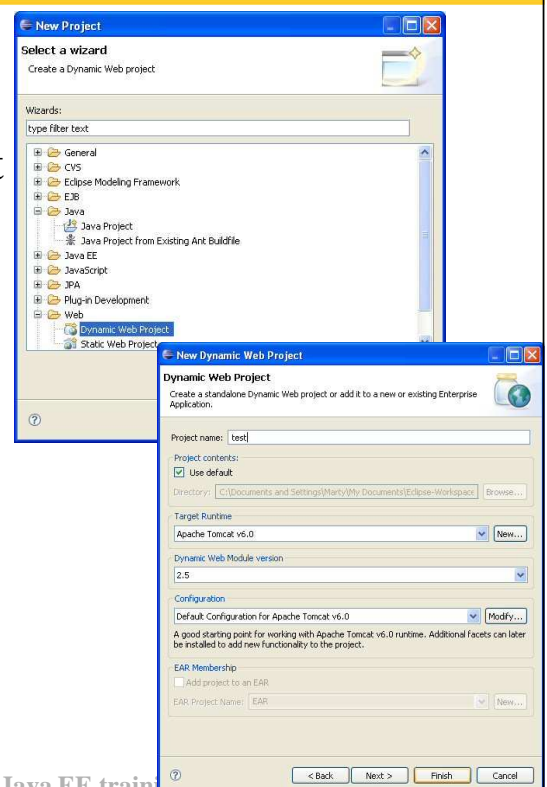


12

Java EE tra

Making Web Apps in Eclipse

- **Make empty project**
 - File → New → Project → Web → Dynamic Web Project
 - Give it a name (e.g., "test")
 - Accept all other defaults
- **Shortcut**
 - If you have made Dynamic Web Project recently in workspace, you can just do File → New → Dynamic Web Project



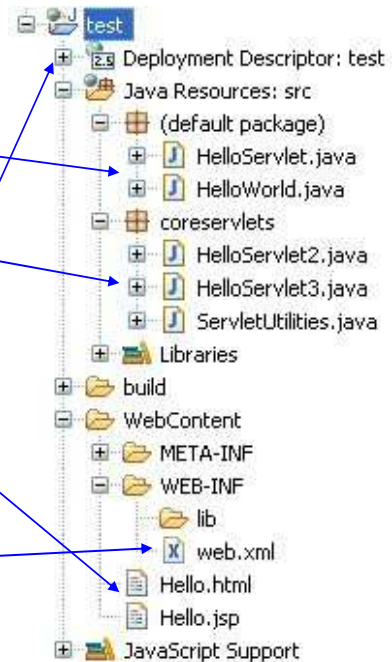
13

Java EE train

Adding Code to Eclipse Projects

• Locations

- src
 - Unpackaged Java code
 - Packages strongly recommended
- src/somePackage
 - Java code in somePackage package
- WebContent
 - Web files (HTML, JavaScript, CSS, JSP, images, etc.)
- WebContent/some-subdirectory
 - Web content in subdirectory
- WebContent/WEB-INF
 - web.xml (will be discussed later)
 - Can also click on "Deployment Descriptor"



• Note

- Can cut/paste or drag/drop files into appropriate locations

14

Java EE training: <http://courses.coreservlets.com>

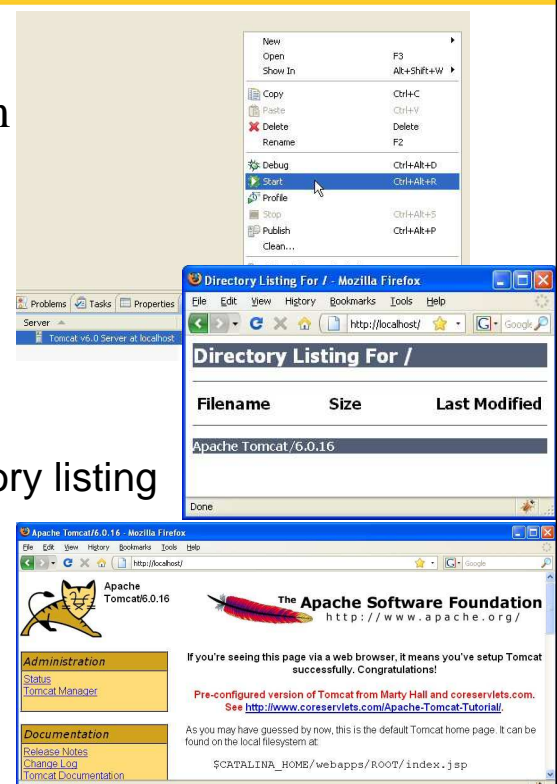
Starting Server in Eclipse

• Start Tomcat

- Select "Servers" tab at bottom
- R-click on Tomcat
- Choose "Start"

• Verify server startup

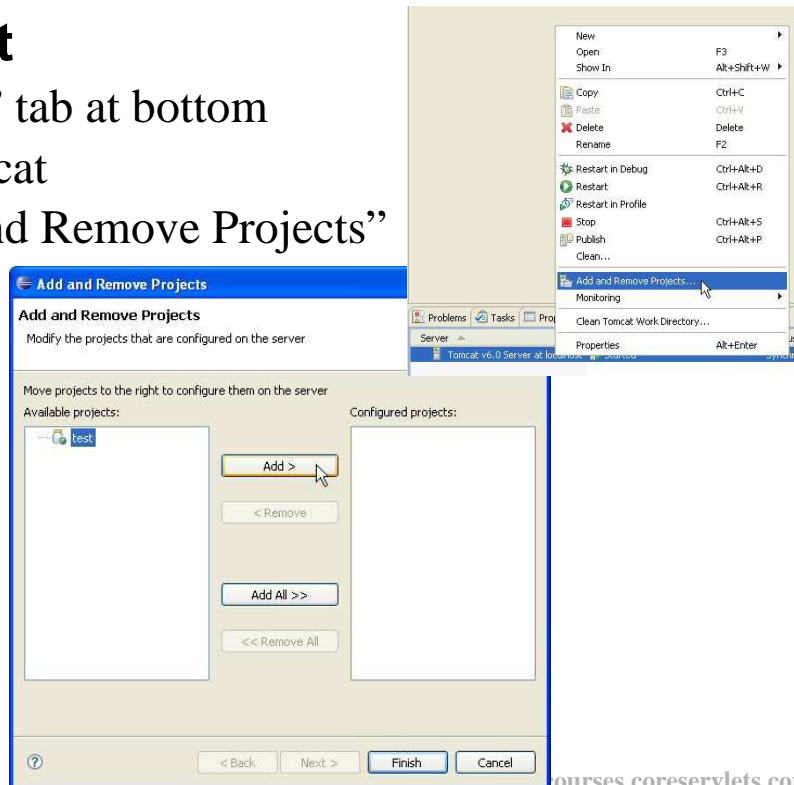
- Open browser
- Enter `http://localhost/`
 - You should see blank directory listing
 - If you want pretty Tomcat welcome page, search for a folder called ROOT in your Eclipse workspace. Copy files from `C:\tomcat-dir\webapps\ROOT` to that folder



15

Deploying App in Eclipse

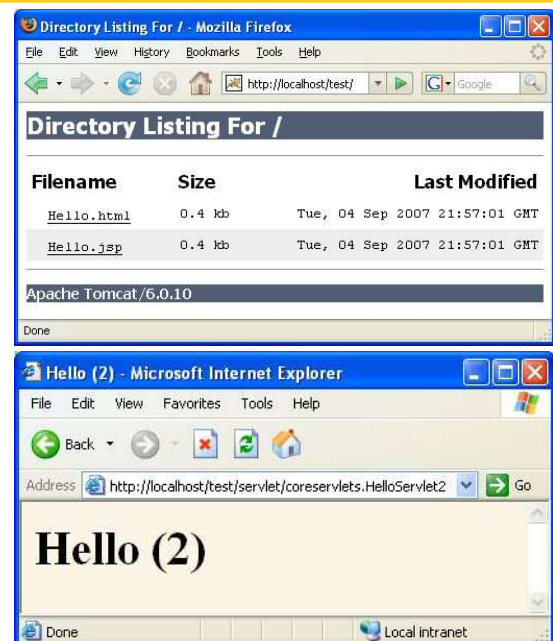
- **Deploy project**
 - Select “Servers” tab at bottom
 - R-click on Tomcat
 - Choose “Add and Remove Projects”
 - Choose project
 - Press “Add”
 - Click “Finish”
- **Restart Server**
 - R-click Tomcat at bottom
 - Restart



16

Testing Deployed Apps in Eclipse

- **Start a browser**
 - Eclipse also has builtin browser, but I prefer to use Firefox or Internet Explorer
- **Test base URL**
 - `http://localhost/test/`
- **Test Web content**
 - `http://localhost/test/Hello.html` (case sensitive!)
 - `http://localhost/test/Hello.jsp`
 - If you used subdirectories
 - `http://localhost/test/some-subdirectory/blah.html`
- **Test servlets**
 - `http://localhost/test/servlet/HelloServlet`
 - `http://localhost/test/servlet/coreservlets.HelloServlet2`
 - Note: custom URLs discussed in next section



17

Defining Custom URLs

- **Java code**

```
package myPackage; ...
public class MyServlet extends HttpServlet { ... }
```

- **web.xml entry (in <web-app...>...</web-app>)**

- Give name to servlet

```
<servlet>
  <servlet-name>MyName</servlet-name>
  <servlet-class>myPackage.MyServlet</servlet-class>
</servlet>
```

- Give address (URL mapping) to servlet

```
<servlet-mapping>
  <servlet-name>MyName</servlet-name>
  <url-pattern>/MyAddress</url-pattern>
</servlet-mapping>
```

- **Resultant URL**

- `http://hostname/webappPrefix/MyAddress`

18

Java EE training: <http://courses.coreservlets.com>

Defining Custom URLs: Example (Assume Eclipse Project is "test")

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://java.sun.com/xml/ns/javaee"
  ...
  version="2.5">

  <!-- Use the URL http://hostname/intro/hi instead
        of http://hostname/intro/servlet/HelloServlet -->
  <servlet>
    <servlet-name>Second Hello Servlet</servlet-name>
    <servlet-class>coreservlets.HelloServlet2</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Second Hello Servlet</servlet-name>
    <url-pattern>/hi2</url-pattern>
  </servlet-mapping>
</web-app>
```

Don't edit this manually.
Should refer to version 2.4
or 2.5 (Tomcat 6 only).

Fully qualified classname.

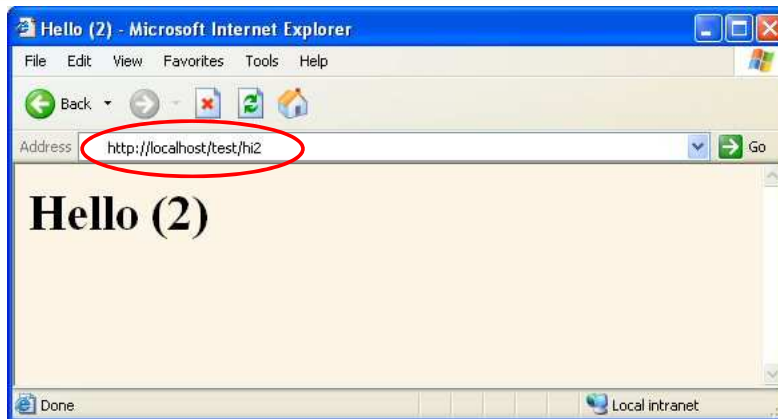
Any arbitrary name.
But must be the same both times.

The part of the URL that comes after the app (project) name.
Should start with a slash.

19

Java EE training: <http://courses.coreservlets.com>

Defining Custom URLs: Result



- **Eclipse details**

- Name of Eclipse project is "test"
- Servlet is in src/coreservlets/HelloServlet2.java
- Deployed by right-clicking on Tomcat, Add and Remove Projects, Add, choosing test project, Finish, right-clicking again, Start

20

Java EE training: <http://courses.coreservlets.com>

Debugging Servlets

- **Use print statements; run server on desktop**
- **Use Apache Log4J**
- **Integrated debugger in IDE**
 - Right-click in left margin in source to set breakpoint (Eclipse)
 - R-click Tomcat and use "Debug" instead of "Start"
- **Look at the HTML source**
- **Return error pages to the client**
 - Plan ahead for missing or malformed data
- **Use the log file**
 - log("message") or log("message", Throwable)
- **Separate the request and response data .**
 - Request: see EchoServer at www.coreservlets.com
 - Response: see WebClient at www.coreservlets.com
- **Make sure browser is not caching**
 - Internet Explorer: use Shift-RELOAD
 - Firefox: use Control-RELOAD
- **Stop and restart the server**

21

Java EE training: <http://courses.coreservlets.com>



Form Data

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

26

Using Form Data

- **HTML form**

- Should have ACTION referring to servlet
 - Use relative URL
 - ACTION="/webAppName/address"
 - ACTION="./address"
- Should have input entries with NAMES
- Should be installed under WebContent

- **Servlet**

- Calls request.getParameter with name as given in HTML
- Return value is entry as entered by end user
- Missing values
 - null if no input element of that name was in form
 - Empty string if form submitted with empty textfield

27

An HTML Form With Three Parameters

`<FORM ACTION="./show-params">` — url-pattern of servlet

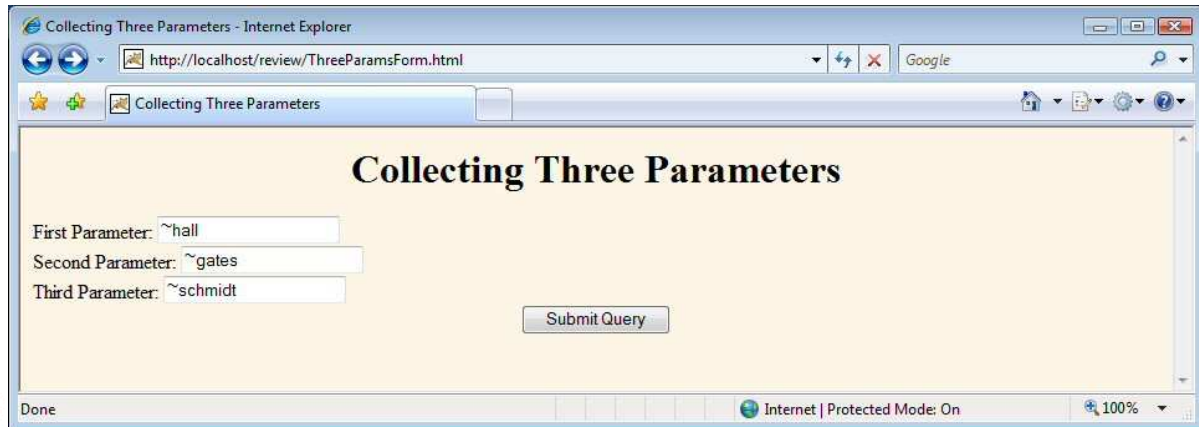
First Parameter: `<INPUT TYPE="TEXT" NAME="param1">
`

Second Parameter: `<INPUT TYPE="TEXT" NAME="param2">
`

Third Parameter: `<INPUT TYPE="TEXT" NAME="param3">
`

`<CENTER><INPUT TYPE="SUBMIT"></CENTER>`

`</FORM>`



- Project name is “review”
- Form installed in WebContent/ThreeParamsForm.html

28

Java EE training: <http://courses.coreservlets.com>

Reading the Three Parameters

```
public class ThreeParams extends HttpServlet {
    public void doGet(HttpServletRequest request,
                     HttpServletResponse response)
        throws ServletException, IOException {
        ...
        out.println(docType +
            "<HTML>\n" +
            "<HEAD><TITLE>"+title + "</TITLE></HEAD>\n" +
            "<BODY BGCOLOR=\"#FDF5E6\">\n" +
            "<H1 ALIGN=\"CENTER\">" + title + "</H1>\n" +
            "<UL>\n" +
            "  <LI><B>param1</B>: "
            + request.getParameter("param1") + "\n" +
            "  <LI><B>param2</B>: "
            + request.getParameter("param2") + "\n" +
            "  <LI><B>param3</B>: "
            + request.getParameter("param3") + "\n" +
            "</UL>\n" +
            "</BODY></HTML>");
    }
}
```

29

Java EE training: <http://courses.coreservlets.com>

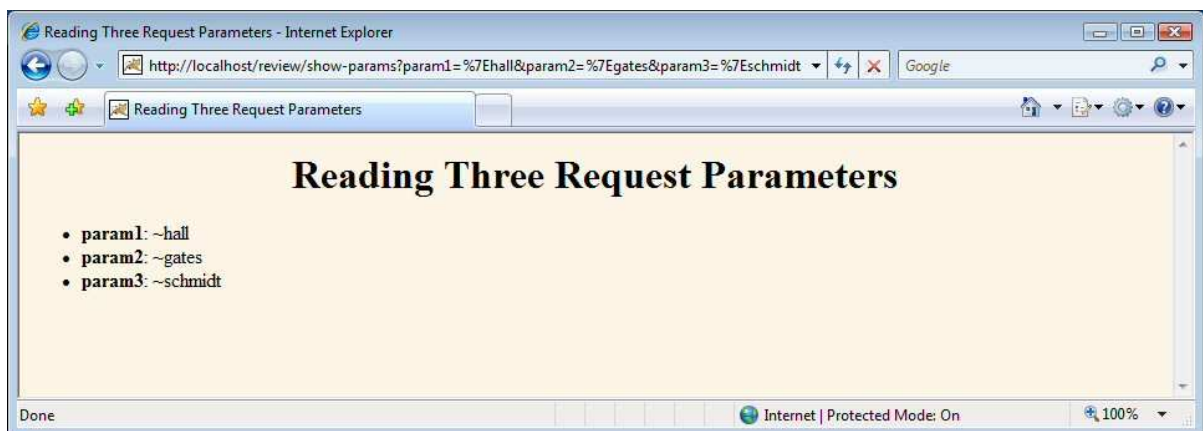
Reading Three Parameters: web.xml

```
...  
<servlet>  
  <servlet-name>Param Servlet</servlet-name>  
  <servlet-class>coreservlets.ThreeParams</servlet-class>  
</servlet>  
<servlet-mapping>  
  <servlet-name>Param Servlet</servlet-name>  
  <url-pattern>/show-params</url-pattern>  
</servlet-mapping>  
...
```

30

Java EE training: <http://courses.coreservlets.com>

Reading Three Parameters: Result



31

Java EE training: <http://courses.coreservlets.com>



JSP Scripting

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

Uses of JSP Constructs

Simple
Application



Complex
Application

- **Scripting elements calling servlet code directly**
- **Scripting elements calling servlet code indirectly (by means of utility classes)**
- **Beans**
- **Servlet/JSP combo (MVC)**
- **MVC with JSP expression language**
- **Custom tags**
- **MVC with beans, custom tags, and a framework like Struts or JSF**

JSP Scripting Design Strategy: Limit Java Code in JSP Pages

- **You have two options**
 - Put 25 lines of Java code directly in the JSP page
 - Put those 25 lines in a separate Java class and put 1 line in the JSP page that invokes it
- **Why is the second option *much* better?**
 - **Development.** You write the separate class in a Java environment (editor or IDE), not an HTML environment
 - **Debugging.** If you have syntax errors, you see them immediately at compile time. Simple print statements can be seen.
 - **Testing.** You can write a test routine with a loop that does 10,000 tests and reapply it after each change.
 - **Reuse.** You can use the same class from multiple pages.

34

Java EE training: <http://courses.coreservlets.com>

JSP Expressions

- **Format**
 - `<%= Java Expression %>`
- **Result**
 - Expression evaluated, converted to String, and placed into HTML page at the place it occurred in JSP page
 - That is, expression placed in `_jspService` inside `out.print`
- **Examples**
 - Current time: `<%= new java.util.Date() %>`
 - Your hostname: `<%= request.getRemoteHost() %>`
- **XML-compatible syntax**
 - `<jsp:expression>Java Expression</jsp:expression>`
 - You cannot mix versions within a single page. You must use XML for *entire* page if you use `jsp:expression`.

35

Java EE training: <http://courses.coreservlets.com>

Predefined Variables

- **request**
 - The HttpServletRequest (1st argument to service/doGet)
- **response**
 - The HttpServletResponse (2nd arg to service/doGet)
- **out**
 - The Writer (a buffered version of type JspWriter) used to send output to the client
- **session**
 - The HttpSession associated with the request (unless disabled with the session attribute of the page directive)
- **application**
 - The ServletContext (for sharing data) as obtained via `getServletContext()`.

36

Java EE training: <http://courses.coreservlets.com>

JSP Scriptlets

- **Format**
 - `<% Java Code %>`
- **Result**
 - Code is inserted verbatim into servlet's `_jspService`
- **Example**
 - `<%
String queryData = request.getQueryString();
out.println("Attached GET data: " + queryData);
%>`
 - `<% response.setContentType("text/plain"); %>`
- **XML-compatible syntax**
 - `<jsp:scriptlet>Java Code</jsp:scriptlet>`

37

Java EE training: <http://courses.coreservlets.com>

JSP Declarations

- **Format**
 - `<%! Java Code %>`
- **Result**
 - Code is inserted verbatim into servlet's class definition, outside of any existing methods
- **Examples**
 - `<%! private int someField = 5; %>`
 - `<%! private void someMethod(...) {...} %>`
- **Design consideration**
 - Fields are clearly useful. For methods, it is usually better to define the method in a separate Java class.
- **XML-compatible syntax**
 - `<jsp:declaration>Java Code</jsp:declaration>`

38

© 2008 Marty Hall



JSP Pages with XML Syntax

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

39

Why Two Versions?

- **Classic syntax is not XML-compatible**
 - `<%= ... %>`, `<% ... %>`, `<%! ... %>` are illegal in XML
 - HTML 4 is not XML compatible either
 - So, you cannot use XML editors like XML Spy
- **You might use JSP in XML environments**
 - To build xhtml pages
 - To build regular XML documents
 - You can use classic syntax to build XML documents, but it is sometimes easier if you are working in XML to start with
 - For Web services
 - For Ajax applications
- **So, there is a second syntax**
 - Following XML rules

40

Java EE training: <http://courses.coreservlets.com>

XML Syntax for Generating XHTML Files (somefile.jsp x)

```
<?xml version="1.0" encoding="UTF-8" ?>
<html xmlns:jsp="http://java.sun.com/JSP/Page">
<jsp:output
  omit-xml-declaration="true"
  doctype-root-element="html"
  doctype-public="-//W3C//DTD XHTML 1.0 Transitional//EN"
  doctype-system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd" />
<jsp:directive.page contentType="text/html"/>
<head><title>Some Title</title></head>
<body bgcolor="#fdf5e6">
Body
</body></html>
```

The jsp namespace is required if you use jsp:blah commands. You can use other namespaces for other custom tag libraries.

Needed because of Internet Explorer bug where xhtml pages that have the XML declaration at the top run in quirks mode.

Builds DOCTYPE line.

For JSP pages in XML syntax, default content type is text/xml.

Normal xhtml content, plus JSP commands that use jsp:blah syntax, plus JSP custom tag libraries.

41

Java EE training: <http://courses.coreservlets.com>

XML Syntax for Generating Regular XML Files (somefile.jspx)

```
<?xml version="1.0" encoding="UTF-8" ?>  
<your-root-element xmlns:jsp="http://java.sun.com/JSP/Page">  
  <your-tag1>foo</your-tag1>  
  <your-tag2>bar</your-tag2>  
</your-root-element>
```

- **Uses**

- When you are sending to client that expects real XML
 - Ajax
 - Web services
 - Custom clients
- Note
 - You can omit the xmlns declaration if you are not using any JSP tags. But then you could just use .xml extension.

XML Syntax for Generating HTML 4 Files (somefile.jspx)

- **Many extra steps required**

- Enclose the entire page in jsp:root
- Enclose the HTML in CDATA sections
 - Between <![CDATA[and]]>
 - Because HTML 4 does not obey XML rules
- Usually not worth the bother

Sample HTML 4 Page: Classic Syntax (sample.jsp)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD ...">
<HTML>
<HEAD><TITLE>Sample (Classic Syntax)</TITLE></HEAD>
<BODY BGCOLOR="#FDF5E6">
<CENTER>
<H1>Sample (Classic Syntax)</H1>

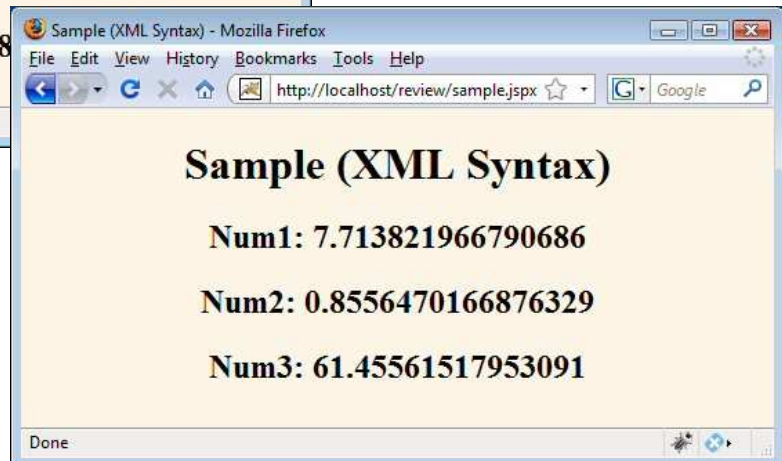
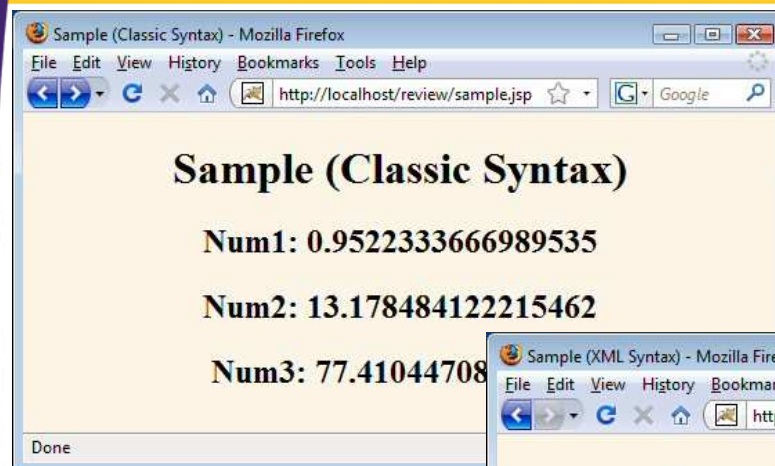
<H2>Num1: <%= Math.random()*10 %></H2>
<% double num2 = Math.random()*100; %>
<H2>Num2: <%= num2 %></H2>
<%! private double num3 = Math.random()*1000; %>
<H2>Num3: <%= num3 %></H2>

</CENTER>
</BODY></HTML>
```

Sample XHTML Page: XML Syntax (sample.jspx)

```
<?xml version="1.0" encoding="UTF-8" ?>
<html xmlns:jsp="http://java.sun.com/JSP/Page">
<jsp:output
  omit-xml-declaration="true"
  doctype-root-element="html"
  doctype-public="-//W3C//DTD ..."
  doctype-system="http://www.w3.org...dtd" />
<jsp:directive.page contentType="text/html"/>
<head><title>Sample (XML Syntax)</title></head>
<body bgcolor="#fdf5e6">
<div align="center">
<h1>Sample (XML Syntax)</h1>
<h2>Num1: <jsp:expression>Math.random()*10</jsp:expression></h2>
<jsp:scriptlet>
double num2 = Math.random()*100;
</jsp:scriptlet>
<h2>Num2: <jsp:expression>num2</jsp:expression></h2>
<jsp:declaration>
private double num3 = Math.random()*1000;
</jsp:declaration>
<h2>Num3: <jsp:expression>num3</jsp:expression></h2>
</div></body></html>
```

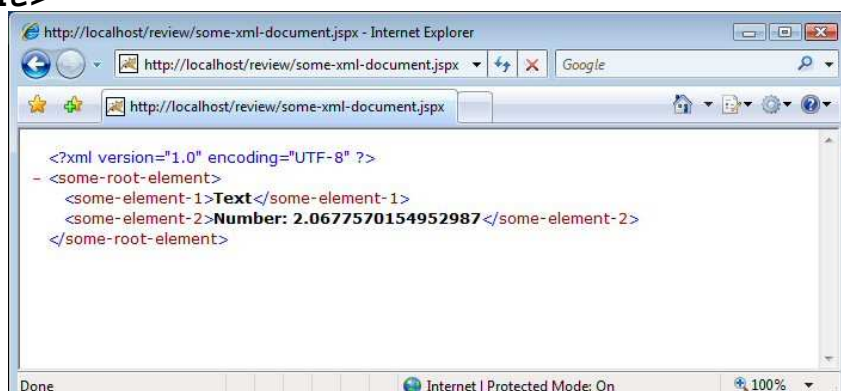
Sample Pages: Results



46

XML Document Generated with XML Syntax

```
<?xml version="1.0" encoding="UTF-8" ?>
<some-root-element
  xmlns:jsp="http://java.sun.com/JSP/Page">
  <some-element-1>Text</some-element-1>
  <some-element-2>
    Number:
    <jsp:expression>Math.random()*10</jsp:expression>
  </some-element-2>
</some-root-element>
```



47



jsp:include

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

48

Including Files at Request Time: jsp:include

- **Format**
 - `<jsp:include page="Relative URL" />`
- **Purpose**
 - To reuse JSP, HTML, or plain text content
 - To permit updates to the included content without changing the main JSP page(s)
- **Notes**
 - JSP content cannot affect main page: only *output* of included JSP page is used
 - Don't forget that trailing slash
 - Relative URLs that starts with slashes are interpreted relative to the Web app, not relative to the server root.
 - You are permitted to include files from WEB-INF

49

jsp:include Example: A News Headline Page (Main Page)

```
...
<BODY>
<TABLE BORDER=5 ALIGN="CENTER">
  <TR><TH CLASS="TITLE">
    What's New at JspNews.com</TABLE>
<P>
Here is a summary of our three
most recent news stories:
<OL>
  <LI><jsp:include page="/WEB-INF/includes/item1.jsp" />
  <LI><jsp:include page="/WEB-INF/includes/item2.jsp" />
  <LI><jsp:include page="/WEB-INF/includes/item3.jsp" />
</OL>
</BODY></HTML>
```

A News Headline Page, Continued (First Included Page)

```
<B>Bill Gates acts humble.</B> In a startling
and unexpected development, Microsoft big wig
Bill Gates put on an open act of humility
yesterday.
<A HREF="http://www.microsoft.com/Never.html">
More details...</A>
```

- Note that the page is *not* a complete HTML document; it has only the tags appropriate to the place that it will be inserted

A News Headline Page: Result



52

Java EE training: <http://courses.coreservlets.com>

© 2008 Marty Hall



MVC

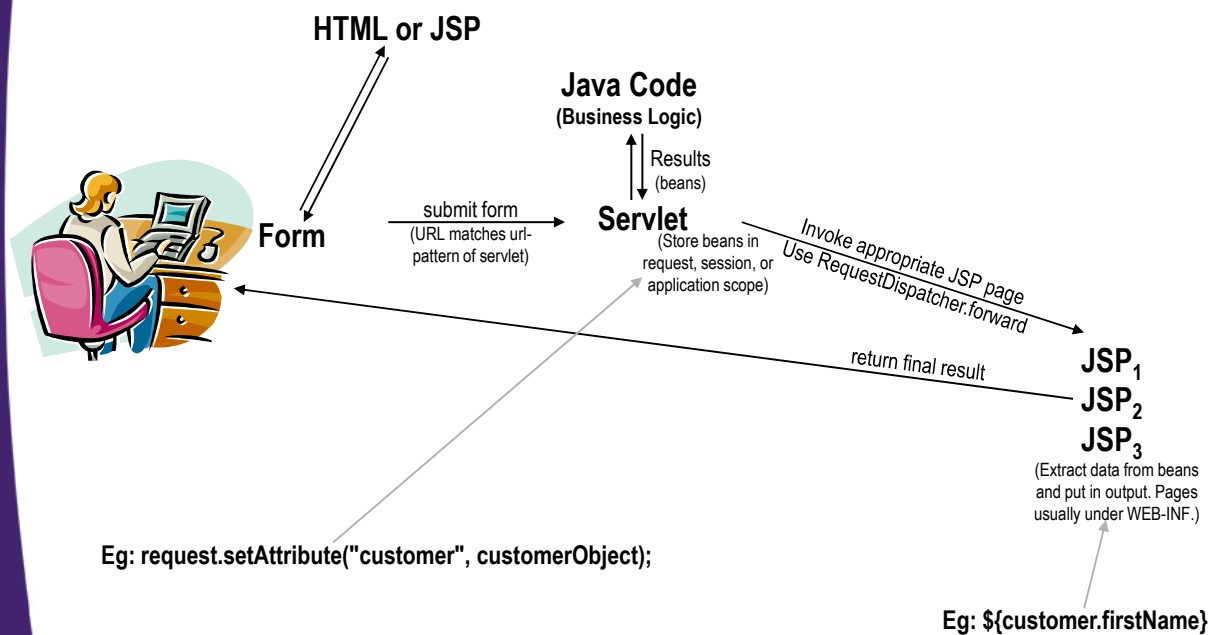
Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

53

MVC Flow of Control



54

Java EE training: <http://courses.coreservlets.com>

Simple MVC Example: Request-Scoped Data

- **Goal**
 - Display a random number to the user
- **Type of sharing**
 - Each request should result in a new number, so request-based sharing is appropriate.

55

Java EE training: <http://courses.coreservlets.com>

Request-Based Sharing: Bean

```
package coreservlets;

public class NumberBean {
    private double num = 0;

    public NumberBean(double number) {
        setNumber(number);
    }

    public double getNumber() {
        return(num);
    }

    public void setNumber(double number) {
        num = number;
    }
}
```

56

Java EE training: <http://courses.coreservlets.com>

Request-Based Sharing: Servlet

```
public class RandomNumberServlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
        throws ServletException, IOException {
        NumberBean bean =
            RanUtils.getRandomNum(request.getParameter("range"));
        request.setAttribute("randomNum", bean);
        String address = "/WEB-INF/mvc-sharing/RandomNum.jsp";
        RequestDispatcher dispatcher =
            request.getRequestDispatcher(address);
        dispatcher.forward(request, response);
    }
}
```

57

Java EE training: <http://courses.coreservlets.com>

Request-Based Sharing: Business Logic

```
public class RanUtils {
    public static NumberBean getRandomNum(String rangeString) {
        double range;
        try {
            range = Double.parseDouble(rangeString);
        } catch (Exception e) {
            range = 10.0;
        }
        return(new NumberBean(Math.random() * range));
    }
}
```

Request-Based Sharing: URL Pattern (web.xml)

```
...
<servlet>
    <servlet-name>RandomNumberServlet</servlet-name>
    <servlet-class>
        coreservlets.RandomNumberServlet
    </servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>RandomNumberServlet</servlet-name>
    <url-pattern>/random-number</url-pattern>
</servlet-mapping>
...
```

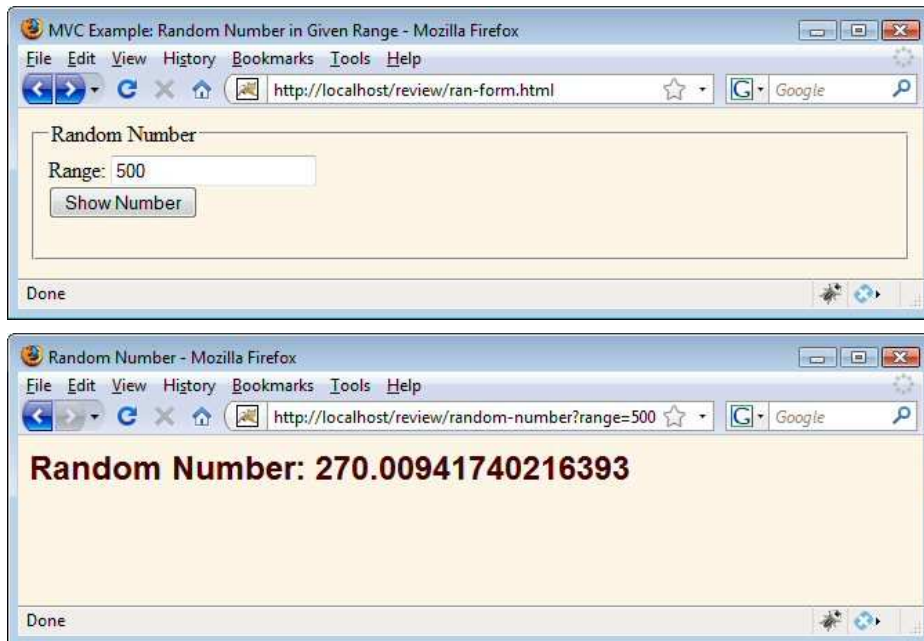
Request-Based Sharing: Input Form

```
...
<fieldset>
  <legend>Random Number</legend>
  <form action="./random-number">
    Range: <input type="text" name="range"><br/>
    <input type="submit" value="Show Number">
  </form>
</fieldset>
...
```

Request-Based Sharing: Results Page

```
...
<body>
<h2>Random Number:  $\${randomNum.number}$ </h2>
</body></html>
```

Request-Based Sharing: Results



62

Java EE training: <http://courses.coreservlets.com>

Summary

- **Set up Java 6, Tomcat, and Eclipse**
 - <http://www.coreservlets.com/apache-tomcat-tutorial/eclipse.html>
- **Give custom URLs to all servlets**
 - Use servlet, servlet-mapping, and url-pattern in web.xml
- **Forms**
 - Use relative URLs for ACTION.
 - Read parameters with `request.getParameter`
- **JSP Scripting**
 - If you use scripting, put most Java code in regular classes
- **MVC**
 - Very widely applicable approach.
 - Consider using it in many (most?) applications

63

Java EE training: <http://courses.coreservlets.com>



Questions?

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces/Facelets, Ajax, GWT, Spring, Hibernate, Java 5 & 6.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.