



## Integrating Servlets and JSP: The MVC Architecture

Core Servlets & JSP book: [www.coreservlets.com](http://www.coreservlets.com)  
 More Servlets & JSP book: [www.moreservlets.com](http://www.moreservlets.com)  
 Servlet and JSP Training Courses: [courses.coreservlets.com](http://courses.coreservlets.com)

Slides © Marty Hall, <http://www.coreservlets.com>, book © Sun Microsystems Press

## Agenda

- **Reasons to combine servlets and JSP**
- **Approach to integration**
- **Dispatching requests**
- **Storing data for later retrieval**
- **Example 1: an on-line travel agent**
- **Example 2: an on-line boat store**
- **Including requests: showing raw servlet and JSP output**

www.coreservlets.com

## 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**
- **Custom tags**
- **Servlet/JSP combo (MVC), with beans and possibly custom tags**

www.coreservlets.com

## Why Combine Servlets & JSP?

- **Typical picture: use JSP to make it easier to develop and maintain the HTML content**
  - For simple dynamic code, call servlet code from scripting elements
  - For slightly more complex applications, use custom classes called from scripting elements
  - For moderately complex applications, use beans and custom tags
- **But, that's not enough**
  - For complex processing, starting with JSP is awkward
  - Despite the ease of separating the real code into separate classes, beans, and custom tags, the assumption behind JSP is that a *single* page gives a *single* basic look

www.coreservlets.com

## Possibilities for Handling a Single Request

- **Servlet only**
  - Output is a binary type. E.g.: an image
  - No output. E.g.: you are doing forwarding or redirection as in Search Engine example.
  - Format/layout of page is highly variable. E.g.: portal.
- **JSP only**
  - Output is mostly character data. E.g.: HTML
  - Format/layout mostly fixed.
- **Combination**
  - A single request will result in multiple substantially different-looking results.
  - Complicated data processing, but relatively fixed layout.
- **These apply to a *single* request**
  - You still use both servlets and JSP within your *overall* application.

www.coreservlets.com

## Approach

- **Joint servlet/JSP process:**
  - Original request is answered by a servlet
  - Servlet processes request data, does database lookup, business logic, etc.
  - Results are placed in beans
  - Request is forwarded to a JSP page to format result
  - Different JSP pages can be used to handle different types of presentation
- **Often called the "MVC" (Model View Controller) or "Model 2" approach to JSP**
- **Formalized in Apache Struts Framework**
  - <http://jakarta.apache.org/struts/>

www.coreservlets.com

## Implementing MVC

- **The important thing is the idea**
  - Syntax not complicated
- **We already know**
  - How to extract previously-stored data in a JSP page
    - Use `jsp:useBean` with the scope attribute
- **Two pieces of syntax we don't yet know**
  - How does a servlet invoke a JSP page?
  - How does a servlet store data where it can be retrieved by
    - `jsp:useBean` with scope="request"
    - `jsp:useBean` with scope="session"
    - `jsp:useBean` with scope="application"

MVC Architecture

www.coreservlets.com

## Dispatching Requests from Servlets to JSP Pages

- **First, call the `getRequestDispatcher` method of `ServletContext`**
  - Supply URL relative to server or Web application root
  - Example:

```
String url = "/presentations/presentation1.jsp";
RequestDispatcher dispatcher =
    getServletContext().getRequestDispatcher(url);
```
- **Second**
  - Call forward to completely transfer control to destination page (no communication with client in between, as there is with `response.sendRedirect()`)
    - This is the normal approach with MVC
  - Call include to insert output of destination page and then continue on

MVC Architecture

www.coreservlets.com

## Forwarding Requests: Example Code

```
public void doGet(HttpServletRequest request,
                 HttpServletResponse response)
    throws ServletException, IOException {
    String operation = request.getParameter("operation");
    if (operation == null) {
        operation = "unknown";
    }
    if (operation.equals("operation1")) {
        gotoPage("/operations/presentation1.jsp",
                request, response);
    } else if (operation.equals("operation2")) {
        gotoPage("/operations/presentation2.jsp",
                request, response);
    } else {
        gotoPage("/operations/unknownRequestHandler.jsp",
                request, response);
    }
}
```

MVC Architecture

www.coreservlets.com

## Forwarding Requests: Example Code (Continued)

```
private void gotoPage(String address,
                     HttpServletRequest request,
                     HttpServletResponse response)
    throws ServletException, IOException {
    RequestDispatcher dispatcher =
        getServletContext().getRequestDispatcher(address);
    dispatcher.forward(request, response);
}
```

MVC Architecture

www.coreservlets.com

## Reminder: JSP useBean Scope Alternatives

- **request**
  - `<jsp:useBean id="..." class="..." scope="request" />`
- **session**
  - `<jsp:useBean id="..." class="..." scope="session" />`
- **application**
  - `<jsp:useBean id="..." class="..." scope="application" />`
- **page**
  - `<jsp:useBean id="..." class="..." scope="page" />`
  - or just
  - `<jsp:useBean id="..." class="..." />`
  - This scope is not used in MVC (Model 2) architecture

MVC Architecture

www.coreservlets.com

## Storing Data for Later Use: The Servlet Request

- **Purpose**
  - Storing data that servlet looked up and that JSP page will use only in this request.
- **Servlet syntax to store data**

```
SomeClass value = new SomeClass(...);
request.setAttribute("key", value);
// Use RequestDispatcher to forward to JSP
```
- **JSP syntax to retrieve data**

```
<jsp:useBean
    id="key"
    class="somepackage.SomeClass"
    scope="request" />
```

MVC Architecture

www.coreservlets.com

## Storing Data for Later Use: The Servlet Request (Variation)

- **Purpose**
  - Storing data that servlet looked up and that JSP page will use only in this request.
- **Servlet syntax to store data**
  - Add new request parameters to servlet request

```
String address = "/path/resource.jsp?newParam=value";
RequestDispatcher dispatcher =
    getServletContext().getRequestDispatcher(address);
dispatcher.forward(request, response);
```
- **JSP syntax to retrieve data**
  - No `useBean` syntax. However, recall that request parameters can be accessed without explicit Java code by means of `jsp:setProperty`.

MVC Architecture

www.coreservlets.com

## Storing Data for Later Use: The Session Object

- **Purpose**
  - Storing data that servlet looked up and that JSP page will use in this request and in later requests from same client.
- **Servlet syntax to store data**

```
SomeClass value = new SomeClass(...);
HttpSession session =
    request.getSession(true);
session.setAttribute("key", value);
// Use RequestDispatcher to forward to JSP
```
- **JSP syntax to retrieve data**

```
<jsp:useBean
    id="key"
    class="somepackage.SomeClass"
    scope="session" />
```

MVC Architecture

www.coreservlets.com

## Storing Data for Later Use: The Servlet Context

- **Purpose**
  - Storing data that servlet looked up and that JSP page will use in this request and in later requests from *any* client.
- **Servlet syntax to store data**

```
SomeClass value = new SomeClass(...);
getServletContext().setAttribute("key",
    value);
// Use RequestDispatcher to forward to JSP
```
- **JSP syntax to retrieve data**

```
<jsp:useBean
    id="key"
    class="somepackage.SomeClass"
    scope="application" />
```

MVC Architecture

www.coreservlets.com

## Relative URLs in JSP Pages

- **Issue:**
  - Forwarding with a request dispatcher is transparent to the client. *Original* URL is only URL browser knows about.
- **Why does this matter?**
  - What will browser do with tags like the following:

```
<IMG SRC="foo.gif" ...>
<LINK REL=STYLESHEET
    HREF="JSP-Styles.css"
    TYPE="text/css">
<A HREF="bar.jsp">...</A>
```
  - Answer: browser treats them as relative to  *servlet URL*
- **Simplest solution:**
  - Use URLs that begin with a slash

MVC Architecture

www.coreservlets.com

## MVC Example 1: An On-Line Travel Agent



MVC Architecture

www.coreservlets.com

## MVC Example 1: An On-Line Travel Agent

- **All requests include**
  - Email address, password, trip origin, trip destination, start date, and end date
- **Original request answered by servlet**
  - Looks up real name, address, credit card information, frequent flyer data, etc., using email address and password as key. *Data stored in session object.*
- **Depending on what button user pressed, request forwarded to:**
  - Page showing available flights, times, and costs
  - Page showing available hotels, features, and costs
  - Rental car info, edit customer data, error handler

MVC Architecture

www.coreservlets.com

## An On-Line Travel Agent: Servlet Code

```
public void doPost(HttpServletRequest request,
                  HttpServletResponse response)
    ...// Store data in TravelCustomer bean called "customer"
    HttpSession session = request.getSession(true);
    session.setAttribute("customer", customer);
    if (request.getParameter("flights") != null) {
        gotoPage("/travel/BookFlights.jsp",
                request, response);
    } else if ...
}
private void gotoPage(String address,
                    HttpServletRequest request,
                    HttpServletResponse response)
    throws ServletException, IOException {
    RequestDispatcher dispatcher =
        getServletContext().getRequestDispatcher(address);
    dispatcher.forward(request, response);
}
}
```

MVC Architecture

www.coreservlets.com

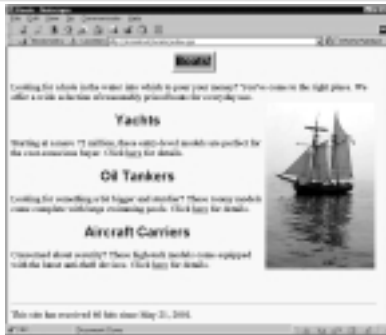
## An On-Line Travel Agent: JSP Code (Flight Page)

```
<BODY>
<H1>Best Available Flights</H1>
<CENTER>
<jsp:useBean id="customer"
             class="coreservlets.TravelCustomer"
             scope="session" />
Finding flights for
<jsp:getProperty name="customer"
                 property="fullName" />
<P>
<jsp:getProperty name="customer" property="flights" />
...
```

MVC Architecture

www.coreservlets.com

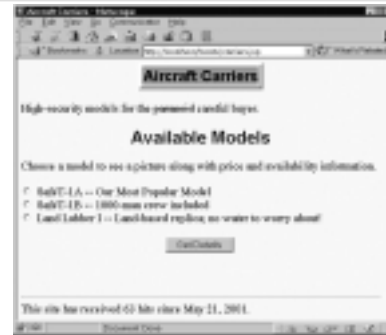
## MVC Example 2: An Online Boat Store



MVC Architecture

www.coreservlets.com

## MVC Example 2: An Online Boat Store



MVC Architecture

www.coreservlets.com

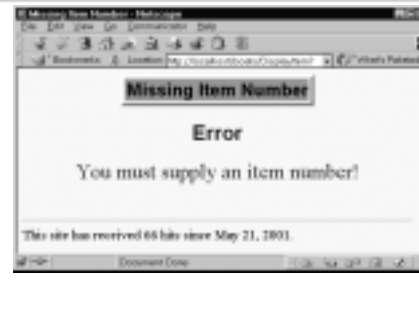
## MVC Example 2: An Online Boat Store



MVC Architecture

www.coreservlets.com

## MVC Example 2: An Online Boat Store



MVC Architecture

www.coreservlets.com

## MVC Example 2: An Online Boat Store



MVC Architecture

www.coreservlets.com

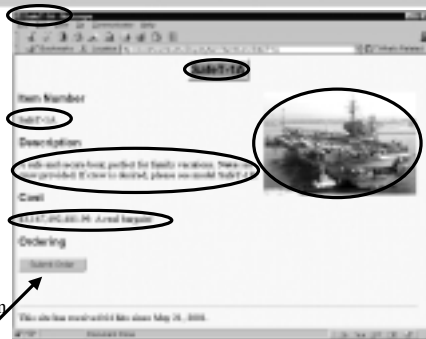
## MVC Example 2: Servlet Code

```
public class ShowItem extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        String itemNum = request.getParameter("itemNum");
        String destination;
        if (itemNum == null) {
            destination = "/MissingItem.jsp";
        } else {
            destination = "/ShowItem.jsp";
            ItemTable shipTable = ShipTable.getShipTable();
            SimpleItem item = shipTable.getItem(itemNum);
            request.setAttribute("item", item);
        }
        RequestDispatcher dispatcher =
            getServletContext().getRequestDispatcher(destination);
        dispatcher.forward(request, response);
    }
}
```

MVC Architecture

www.coreservlets.com

## MVC Example 2: An Online Boat Store



Hidden  
Field

MVC Architecture

www.coreservlets.com

## MVC Example 2: JSP Code (ShowItem.jsp)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
...
<jsp:useBean id="item"
    class="moreservlets.SimpleItem"
    scope="request" />
<TITLE><jsp:getProperty name="item" property="itemNum" />
</TITLE>
...
<TABLE BORDER=5 ALIGN="CENTER">
  <TR><TH CLASS="TITLE">
    <jsp:getProperty name="item" property="itemNum" /></TH>
  </TR>
  <TR>
    <TD>
      <IMG SRC="<jsp:getProperty name='item' property='imageUrl' />"
        ALIGN="RIGHT">
    </TD>
  </TR>
  <TR>
    <TD><H3>Item Number</H3>
    <jsp:getProperty name="item" property="itemNum" />
  </TD>
  <TD><H3>Description</H3>
    <jsp:getProperty name="item" property="description" />
  </TD>
</TR>
</TABLE>
```

MVC Architecture

www.coreservlets.com

## MVC Example 2: JSP Code (ShowItem.jsp Cont.)

```
<H3>Cost</H3>
<jsp:getProperty name="item" property="costString" />.
A real bargain!

<H3>Ordering</H3>
<FORM ACTION="DisplayPurchases">
  <INPUT TYPE="HIDDEN" NAME="itemNum"
    VALUE="<jsp:getProperty name='item'
      property='itemNum' />">
  <INPUT TYPE="SUBMIT" VALUE="Submit Order">
</FORM>

<%@ taglib uri="/WEB-INF/tlds/count-taglib.tld"
  prefix="boats" %>
<boats:count />
</BODY>
</HTML>
```

MVC Architecture

www.coreservlets.com

## MVC Example 2: Bean Code (SimpleItem.java)

```
public class SimpleItem {
    private String itemNum = "Missing item number";
    private String description = "Missing description";
    private String imageUrl = "Missing image URL";
    private double cost;
    private NumberFormat formatter =
        NumberFormat.getCurrencyInstance();

    public SimpleItem(String itemNum,
        String description,
        String imageUrl,
        double cost) {
        setItemNum(itemNum);
        setDescription(description);
        setImageURL(imageURL);
        setCost(cost);
    }

    public SimpleItem() {}
}
```

MVC Architecture

www.coreservlets.com

## Forwarding Requests from JSP Pages -- `jsp:forward`

- You usually forward from a servlet to a JSP page, but you can also forward from JSP

```
<% String destination;
   if (Math.random() > 0.5) {
       destination = "/examples/page1.jsp";
   } else {
       destination = "/examples/page2.jsp";
   }
%>
<jsp:forward page="<%= destination %>" />
```

- Question: can you forward from a servlet to another servlet? How do you know?

MVC Architecture

www.coreservlets.com

## Including Pages Instead of Forwarding to Them

- With the `forward` method of `RequestDispatcher`:
  - Control is *permanently* transferred to new page
  - Original page *cannot* generate any output
- With the `include` method of `RequestDispatcher`:
  - Control is *temporarily* transferred to new page
  - Original page *can* generate output before and after the included page
  - Original servlet does not see the output of the included page (for this, see later topic on servlet/JSP filters)
  - Useful for portals: JSP presents pieces, but pieces arranged in different orders for different users

MVC Architecture

www.coreservlets.com

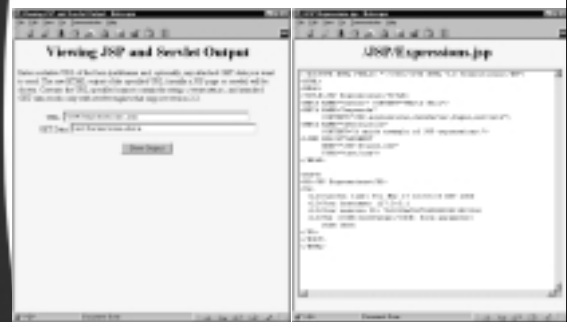
## A Servlet that Shows Raw Servlet and JSP Output

```
out.println(...
    "<TEXTAREA ROWS=30 COLS=70>");
if ((url == null) || (url.length() == 0)) {
    out.println("No URL specified.");
} else {
    // Attaching data works only in version 2.2.
    String data = request.getParameter("data");
    if ((data != null) && (data.length() > 0)) {
        url = url + "?" + data;
    }
    RequestDispatcher dispatcher =
        getServletContext().getRequestDispatcher(url);
    dispatcher.include(request, response);
}
out.println("</TEXTAREA>\n" +
    ...);
```

MVC Architecture

www.coreservlets.com

## A Servlet that Shows Raw Servlet and JSP Output



MVC Architecture

www.coreservlets.com

## Summary

- Use MVC (Model 2) approach when:
  - One submission will result in more than one basic look
  - Several pages have substantial common processing
- Architecture
  - A servlet answers the original request
  - Servlet does the real processing & stores results in beans
    - Beans stored in `HttpServletRequest`, `HttpSession`, or `ServletContext`
  - Servlet forwards to JSP page via forward method of `RequestDispatcher`
  - JSP page reads data from beans by means of `jsp:useBean` with appropriate scope (request, session, or application)

MVC Architecture

www.coreservlets.com



## Questions?

Core Servlets & JSP book: [www.coreservlets.com](http://www.coreservlets.com)  
More Servlets & JSP book: [www.moreservlets.com](http://www.moreservlets.com)  
Servlet and JSP Training Courses: [courses.coreservlets.com](http://courses.coreservlets.com)

Slides © Marty Hall, <http://www.coreservlets.com>, book © Sun Microsystems Press

## More Information

- **Source code for all examples**
  - <http://www.coreservlets.com>
- **Servlet/JSP Training Courses**
  - <http://courses.coreservlets.com>
- **Core Servlets & JSP**
  - <http://www.coreservlets.com>
- **More Servlets & JSP**
  - Sequel to *Core Servlets & JSP*
  - <http://www.moreservlets.com>
- **Servlet home page**
  - <http://java.sun.com/products/servlet/>
- **JavaServer Pages home page**
  - <http://java.sun.com/products/jsp/>



MVC Architecture

[www.coreservlets.com](http://www.coreservlets.com)