

Servlets, Java Server Pages, and Taglibs

COMP 342: Programming Methods

09 September 2008

Servlets

- ▶ A *servlet* is a Java application that runs in a (compliant) web server (like Apache Tomcat).
 - ▶ Really, a Java *class* that extends server capabilities.
 - ▶ The web server is often called a *container*.
- ▶ Officially, a servlet is any class that implements the `javax.servlet.Servlet` interface.
 - ▶ For us: extend `javax.servlet.http.HttpServlet` class.
- ▶ `init` method: invoked by the container when the servlet is created.
- ▶ `service` method: invoked by the container when services must be provided.
 - ▶ For us: override `doGet` and `doPost` from `HttpServlet`.

Servlets

So, a typical servlet:

```
public class MyServlet extends HttpServlet {  
    // Static, instance data.  
  
    // Custom initialization, if necessary.  
    public void init() { ... }  
  
    // Handle GET requests.  
    public void doGet(HttpServletRequest req,  
        HttpServletResponse resp) { ... }  
  
    // Delegate POST requests to GET handler.  
    public void doPost(HttpServletRequest req,  
        HttpServletResponse resp) { doGet(req, resp) ; }  
}
```

Servlets

- ▶ Remember: POST requests used to submit data with request. GET requests can do the same!
 - ▶ GET requests encode data in URL; POST requests in body of protocol message.
- ▶ That data is available via `HttpServletRequest.getParameter*` methods.
- ▶ Main job of `doGet`: emit response to client.
 - ▶ Typically Strings that make up an HTML document.

```
public void doGet (...) {  
    PrintWriter pw = resp.getWriter() ;  
  
    pw.println("<html>") ;  
    pw.println("<head>") ;  
    pw.println("<title>Hello, world!</title>") ;  
    ...  
}
```

Servlets

Many more options available.

- ▶ Cookie management.
- ▶ Session management: Servlets can maintain session state.
- ▶ Maintain complex object hierarchies.
- ▶ Access additional resources (e.g., databases).

But ultimately, the servlet is responsible for constructing the response to client request. And that is generally a pain. . . .

Java Server Pages

What is a Java Server Page?

A Java Server Page consists of static content and JSP elements for dynamic content.

- ▶ The static content is usually HTML.
- ▶ JSP elements in `<% ... %>` tags.
- ▶ Can set/access data variables, execute Java code, do specialized processing.
- ▶ The JSP 2.0 *Expression Language* (EL) provides a simple syntax for accessing application data—more on this later.

Java Server Pages

A simple JSP file

```
<html>

<body>
The time is <% java.util.DateFormatter.getDateInstance().
                format(new java.util.Date()) %>.
</body>

</html>
```

The code inside the JSP *scriptlet* element is executed; the result is converted to a string, and the result substituted for the element.

Java Server Pages

What's really going on

JSP's are a special kind of servlet!

- ▶ When a container receives a request for a JSP, it creates a servlet from the JSP.
- ▶ The `doGet` method is constructed as follows:
 - ▶ Non-JSP elements are converted into

```
response.getWriter().print()
```

statements, where `response` is the `HttpServletResponse` parameter.

- ▶ JSP elements are converted into additional source code.

Java Server Pages

So the previous JSP is converted to...

```
public void doGet(..., HttpServletResponse resp) {  
    PrintWriter pw = resp.getWriter() ;  
  
    pw.print("<html>\n") ;  
    pw.print("<body>\n") ;  
    pw.print("The time is ") ;  
    pw.print(java.util.DateFormatter.getDateInstance().  
        format(new java.util.Date()).toString()) ;  
    pw.print(".\n") ;  
    ...  
}
```

Java Server Pages

JSP elements

- ▶ Declarations: `<%! int i = 0 ; %>`. Create instance variables for later use.
- ▶ Expressions: `<%= i %>`. Get the value of an expression. Value is converted to a string.
- ▶ Scriptlets: `<% ... %>`. Any Java code you like; gets inserted directly into the generated servlet `doGet` method.
- ▶ Expression Language expressions: `$ {...}`. Complex arithmetic and boolean expressions. Very useful for accessing application data. JSP 2.0 only. More later.

Connecting servlet data to JSPs

How do we access servlet objects from a JSP?

- ▶ Remember MVC: JSP is a *view* of some application data (the *model*). The servlet is the *controller* here!
- ▶ The model(s) are *Java beans*:
 - ▶ Public class with public no-argument constructor and accessors:

```
public void setXxx(SomeType param) {...}  
public SomeType getXxx() {...}
```
- ▶ One approach to connecting beans to JSPs:
 - ▶ Servlet constructs beans as appropriate in `doGet`.
 - ▶ Servlet makes each bean an *attribute* of the incoming request.
 - ▶ Servlet forwards the request to the JSP.
 - ▶ JSP can access the bean through either `<jsp:useBean>` element or EL expression—we do the latter.

Taglibs

Tag libraries extend the JSP elements

- ▶ Very roughly: taglib defines tags (new elements) and the corresponding Java code to be executed into JSP servlet.
- ▶ All we care about now: *Java Standard Tag Library* (JSTL).
- ▶ Provides much simpler flow-control syntax than scriptlet elements.
- ▶ Especially useful in combination with the MVC pattern and EL expressions.

Web application structure

Filesystem organization for web applications

```
basename/  
  WEB-INF/  
    web.xml  
  classes/  
    [JAR files with correctly-packaged classes]  
    [Unpacked directories corresponding to packages]  
  lib/  
    [Additional resources such as taglibs]  
    [Other content]
```

- ▶ [Other content] should be well-organized into directories!
- ▶ WAR files have WEB-INF, etc. at the top-level.