Application Design Using Java

Lecture 18

Apache Tomcat

- Software Web server
- Implements
  - Java Servlet
  - JavaServer Pages
  - Java WebSocket
  - more
- Provides a "pure Java" HTTP web server environment in which Java code can run
- Open source

Standard Directory Layout

- ***.html, *.jsp, etc.** - The HTML and JSP pages, along with other files that must be visible to the client browser (such as JavaScript, stylesheet files, and images) for your application. In larger applications you may choose to divide these files into a subdirectory hierarchy, but for smaller apps, it is generally much simpler to maintain only a single directory for these files.
- **/WEB-INF/web.xml** - The Web Application Deployment Descriptor for your application. This is an XML file describing the servlets and other components that make up your application, along with any initialization parameters and container-managed security constraints that you want the server to enforce for you.
- **/WEB-INF/classes/** - This directory contains any Java class files (and associated resources) required for your application, including both servlet and non-servlet classes, that are not combined into JAR files. If your classes are organized into Java packages, you must reflect this in the directory hierarchy under /WEB-INF/classes/. For example, a Java class named com.mycompany.mypackage.MyServlet would need to be stored in a file named /WEB-INF/classes/com/mycompany/mypackage/MyServlet.class.
- **/WEB-INF/lib/** - This directory contains JAR files that contain Java class files (and associated resources) required for your application, such as third party class libraries or JDBC drivers.

JavaServer Pages (JSP)

- An extension of the servlet technology
- Contains two types of text
    - Static data (HTML, SVG, XML, etc.)
    - JSP elements, which construct dynamic content
        - Standard syntax
        - XML syntax
- Recommended extension is .jsp

Scripting Elements

- Scriptlets
  - Dynamic content
  - Enclosed in <% … %> tags
- Declarations
  - Declare methods and variables
  - Can be used from any point in the JSP page
  - Enclosed in <%! … %> tags
- Expressions: `<%= scripting-language-expression %>`
  - The value of a scripting language expression
    - Converted into a string
    - Inserted into the data stream returned to the client
  - A semicolon is not allowed, even if the same expression has a semicolon when used within a scriptlet
- Directives
  - Start with <% and end with %>
  - A @ follows the <%

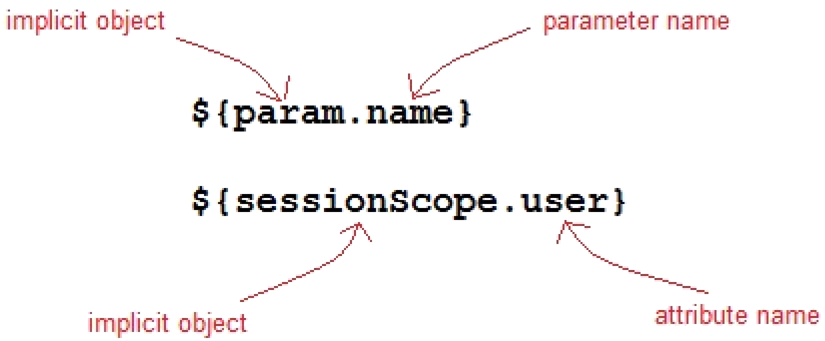
Running a JSP

- Create a directory under webapps for your JSP application
- Add a subdirectory named WEB-INF under the application directory
- Run Tomcat
- Call your JSP application from a Web browser

Built-in JSP Tags (Java Objects)

- request – an HTTP request. A class that implements javax.servlet.http.HttpServletRequest interface.
- response – a response to the request. A class that implements javax.servlet.http.HttpServletResponse interface.
- pageContext – JSP page context (names, tags, attributes). Class javax.servlet.jsp.PageContext.
- session – a session of client communicating with the server. A class that implements javax.servlet.http.HttpSession interface.
- application – an application object. A class that implements javax.servlet.ServletContext interface.
- out – output stream. Class javax.servlet.jsp.JspWriter.
- config – servlet configuration object for the current page. A class that implements javax.servlet.ServletConfig interface.
- page – an object that processes the request for the given page. Object class.

JSP Expression Language (EL)



```
${param.name}

${sessionScope.user}
```

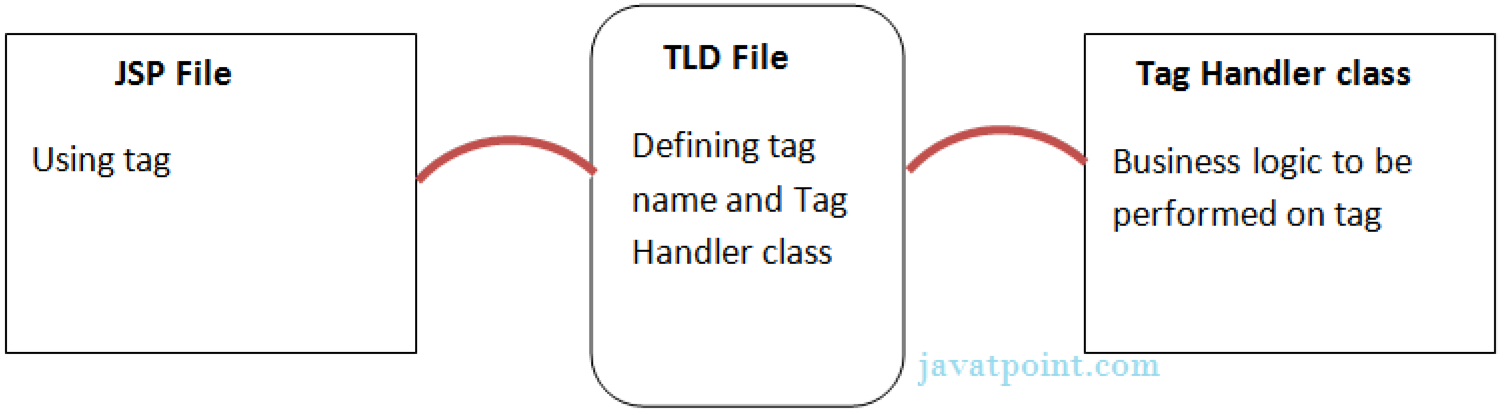
- Syntax: `${expr}`
- Can be mixed with static text/values
- Can be combined with other expressions to form a larger expression
- Usage
  - As attribute values in standard and custom tags,
    e.g., `<jsp:include page="${location}">`
  - To output in HTML tag,
    e.g., `<h1>Welcome ${name}</h1>`

Custom JSP Tags

- A user-defined JSP language element
- Converted to operations on an object called a tag handler
- When the JSP page's servlet is executed those operations are invoked
- Deployment
  - Write Java class
  - Compile Java class:

```
javac -classpath "c:\Program Files\Apache Software Foundation\Tomcat 9.0\lib\*" HelloTag.java
```

  - Write a TLD file
  - Write a JSP


JSP File
Using tag
TLD File
Defining tag name and Tag Handler class
Tag Handler class
Business logic to be performed on tag
javatpoint.com

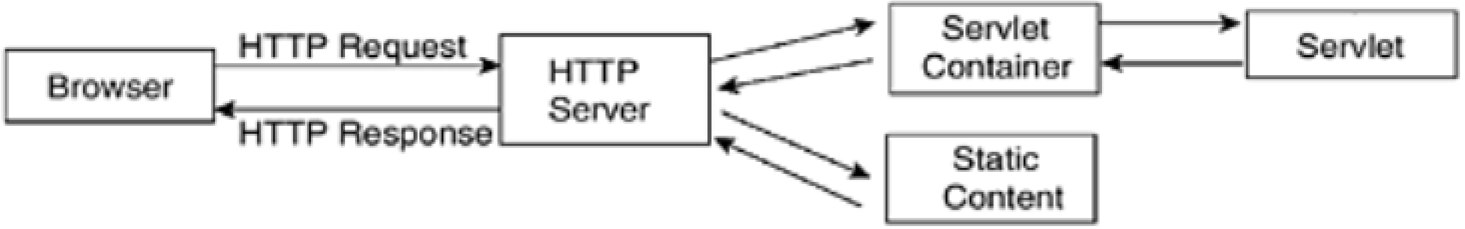
Java Standard Tag Library (JSTL)

- A collection of useful JSP tags
- Encapsulates the core functionality common to many JSP applications
- Requires installation (copying jar files)
- JSTL Tags
  - Core Tags
  - Formatting tags
  - SQL tags
  - XML tags
  - JSTL Functions

JSTL Core Tags

Tag	Description
<c:out>	Like <%= ... >, but for expressions
<c:set>	Sets the result of an expression evaluation in a 'scope'
<c:remove>	Removes a scoped variable (from a particular scope, if specified)
<c:catch>	Catches any Throwable that occurs in its body and optionally exposes it
<c:if>	Simple conditional tag which evaluates its body if the supplied condition is true
<c:choose>	Simple conditional tag that establishes a context for mutually exclusive conditional operations, marked by <when> and <otherwise>
<c:when>	Subtag of <choose> that includes its body if its condition evaluates to 'true'
<c:otherwise>	Subtag of <choose> that follows the <when> tags and runs only if all of the prior conditions evaluated to 'false'
<c:import>	Retrieves an absolute or relative URL and exposes its contents to either the page, a String in 'var', or a Reader in 'varReader'
<c:forEach>	The basic iteration tag, accepting many different collection types and supporting subsetting and other functionality
<c:forTokens>	Iterates over tokens, separated by the supplied delimiters
<c:param>	Adds a parameter to a containing 'import' tag's URL
<c:redirect>	Redirects to a new URL
<c:url>	Creates a URL with optional query parameters

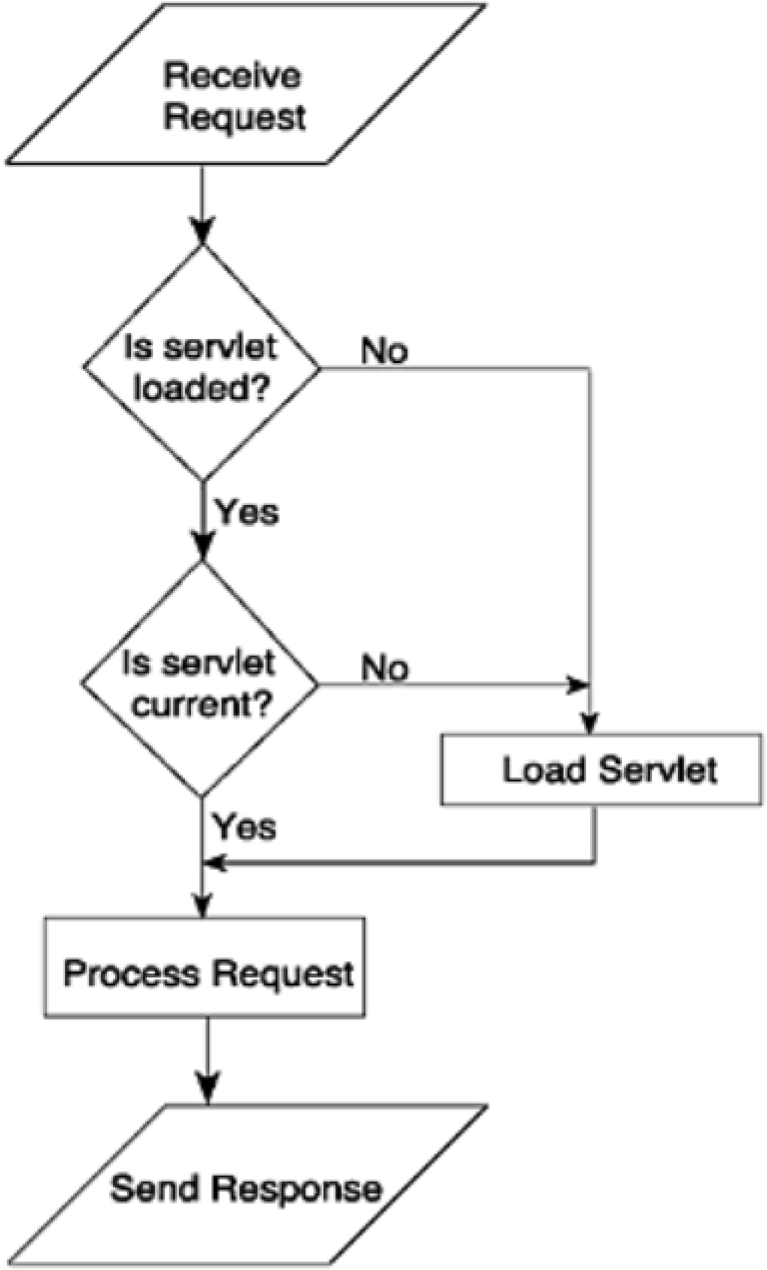
Java Servlet



- Java class
- Can be loaded dynamically
- Run by a special Web server, called a *servlet container*
- Interact with clients via a request-response model based on HTTP
- Must support HTTP/HTTPS
- Can include static content

Servlet Operation

- Loaded by the servlet container the first time the servlet is requested
- User request is forwarded to the servlet
- Servlet processes the user request
- Returns the response to the servlet container
- Servlet container sends the response back to the user



Running a Servlet

- Create a directory structure under Tomcat for your application.
- Write the servlet source code. You need to import the javax.servlet package and the javax.servlet.http package in your source file.
- Compile your source code.
- Create a deployment descriptor.
- Run Tomcat.
- Call your servlet from a Web browser.

//TODO before next lecture:

- Practice Exercises for Lecture 18.