Apache Struts: An MVC Framework

Overview, Installation, and Setup. Struts 1.2 Version.

***Core* Servlets & JSP book: www.coreservlets.com**
***More* Servlets & JSP book: www.moreservlets.com**
Servlet/JSP/Struts/JSF Training: courses.coreservlets.com

For live Struts training, please see JSP/servlet/Struts/JSF 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.

Agenda

- **Understanding Struts**
  - Different views of Struts
  - Advantages of Struts
  - Disadvantages of Struts
- **Setting Up Struts**
  - Downloading and configuring Struts 1.2
  - Testing Struts
  - Setting up Struts applications
  - Accessing Struts documentation
  - Adding Struts to an existing Web application

What is Apache Struts?

- **An MVC Framework?**
  - Struts provides a unified framework for deploying servlet and JSP applications that use the MVC architecture.
- **A Collection of Utilities?**
  - Struts provides utility classes to handle many of the most common tasks in Web application development
- **A Set of JSP Custom Tag Libraries?**
  - Struts provides custom tag libraries for outputting bean properties, generating HTML forms, iterating over various types of data structures, and conditionally outputting HTML.
- **But which is the proper way to view Struts?**
  - The answer depends on what you are going to use it for, but the MVC framework is the most common way of looking at Struts.

Advantages of Struts (vs. MVC Using RequestDispatcher)

- **Centralized file-based configuration**
  - Rather then hard-coding information into Java programs, many Struts values are represented in XML or property files. This loose coupling means that many changes can be made without modifying or recompiling Java code, and that wholesale changes can be made by editing a single file. This approach also lets Java and Web developers focus on their specific tasks (implementing business logic, presenting certain values to clients, etc.) without needing to know about the overall system layout.
- **Form beans**
  - In JSP, you can use property="*" with jsp:setProperty to automatically populate a JavaBean component based on incoming request parameters. Apache Struts extends this capability to Java code and adds in several useful utilities, all of which serve to greatly simplify the processing of request parameters.
- **Bean tags**
  - Apache Struts provides a set of custom JSP tags (bean:write, in particular) that let you easily output the properties of JavaBeans components. Basically, these are concise and powerful variations of the standard jsp:useBean and jsp:getProperty tags.

Advantages of Struts (vs. Standard MVC), Continued

- **HTML tags**
  - Apache Struts provides a set of custom JSP tags to create HTML forms that are associated with JavaBeans components. This bean/form association serves two useful purposes:
    - It lets you get initial form-field values from Java objects.
    - It lets you redisplay forms with some or all previously entered values intact.
- **Form field validation**
  - Apache Struts has builtin capabilities for checking that form values are in the required format. If values are missing or in an improper format, the form can be automatically redisplayed with error messages and with the previously entered values maintained.
    - This validation can be performed on the server (in Java), or both on the server and on the client (in JavaScript).
- **Consistent approach**
  - Struts encourages consistent use of MVC throughout your application.

Disadvantages of Struts (vs. MVC with RequestDispatcher)

- **Bigger learning curve**
  - To use MVC with the standard RequestDispatcher, you need to be comfortable with the standard JSP and servlet APIs. To use MVC with Struts, you have to be comfortable with the standard JSP and servlet APIs and a large and elaborate framework that is almost equal in size to the core system. This drawback is especially significant with smaller projects, near-term deadlines, and less experienced developers; you could spend as much time learning Struts as building your actual system.
- **Worse documentation**
  - Compared to the standard servlet and JSP APIs, Struts has fewer online resources, and many first-time users find the online Apache documentation confusing and poorly organized. There are also fewer books on Apache Struts than on standard servlets and JSP.

Disadvantages of Struts (vs. Standard MVC), Continued

- **Less transparent**
  - With Struts applications, there is a lot more going on behind the scenes than with normal Java-based Web applications. As a result, Struts applications are:
    - Harder to understand
    - Harder to benchmark and optimize
- **Rigid approach**
  - The flip side of the benefit that Struts encourages a consistent approach to MVC is that Struts makes it difficult (but by no means impossible) to use other approaches.

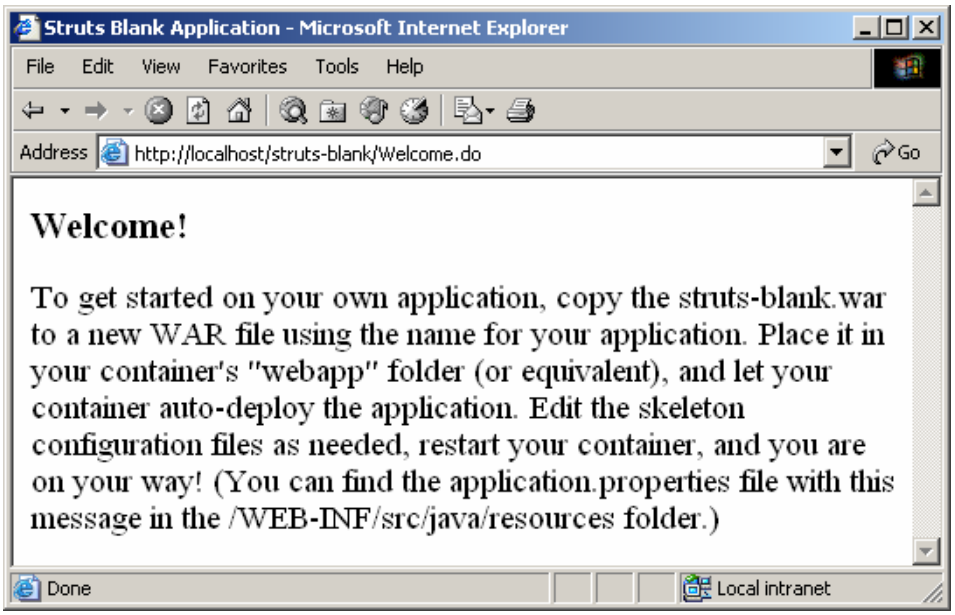
Downloading and Configuring Struts

- **Download the Struts zip file**
  - Start at http://jakarta.apache.org/site/binindex.cgi, or follow the link from http://jakarta.apache.org/struts/.
- **Unzip into a directory of your choice**
  - For example, unzip into C:\jakarta-struts-1.2.4. I'll refer to this location as *struts_install_dir*.
- **Update your CLASSPATH**
  - Add *struts_install_dir*/lib/struts.jar to the CLASSPATH used by your compiler or IDE (not your server).
- **Install an XML parser**
  - *If you have JDK 1.4 or Apache Tomcat, this step is not necessary*, since they already come with an XML parser. But, if you use JDK 1.2 or 1.3 with another server, you might need to obtain the XML parsing libraries. Here are two good sources:
    - http://xml.apache.org/
    - http://java.sun.com/xml/

Testing Struts

- **Install struts-blank.war.**
  - Install the Web application from *struts_install_dir*/webapps/struts-blank.war on your server. For example, with Apache Tomcat, copy *struts_install_dir*/webapps/struts-blank.war to *tomcat_install_dir*/webapps/.
- **Start or restart the server.**
  - Most servers only recognize new Web apps when the server is started.
- **Access http://localhost/struts-blank/.**
  - This URL assumes you are running the server on your desktop and are using port 80. In general, access http://*hostname*:*port*/struts-blank/.

Testing Struts (Results)



Setting Up Struts Applications

- **Option 1 (Good): Copy/rename the struts-blank directory to your development directory.**
  - When you tested your Struts installation, you ran the struts-blank Web app. When you did so, Tomcat and most other servers unpacked the struts-blank.war file into the struts-blank directory. So, copy that directory to whatever location you use for developing custom Web apps, and rename it to whatever app name you choose.
  - For example, if you are using Tomcat and C:\Servlets+JSP is the location you use for development, copy the *tomcat_install_dir*/webapps/struts-blank directory to C:\Servlets+JSP, rename it to struts-test, resulting in C:\Servlets+JSP\struts-test.
  - **Note:** the index.jsp in struts-blank should be replaced with your own version.

Setting Up Struts Applications

- **Option 2 (Good): Unjar struts-blank.war into your development directory.**
  - If your server doesn't automatically unpack WAR files, then even if you ran the struts-blank test described above, you have no regular directory to work with. Besides, some people prefer to unpack the WAR file themselves anyhow. To do so:
    - Copy struts_install_dir/webapps/struts-blank.war to your development directory.
    - Make a new directory (e.g., called struts-test).
    - Unjar struts-blank.war into that directory You can use jar -xvf or a standard zip tool like WinZip or the Windows XP explorer.
    - Whenever you want to test your application, copy struts-test to the server's Web application autodeploy directory (e.g., onto the shortcut to tomcat_install_dir/webapps).
      - Again, remember to replace the index.jsp page in struts-blank

Setting Up Struts Applications

- **Option 3 (Bad): Rename struts-blank and leave it in the server's deployment directory.**
  - It is quite common to work directly in server's deployment directory (e.g., to work directly in *tomcat_install_dir*/webapps/struts-test). However, this approach scales poorly to real applications, makes it harder to test changes (you have no stable working version), and doesn't support your "real" deployment server anyhow (which is almost certainly not your desktop machine). If you don't have a good deployment system using your IDE or ant scripts, you can easily make a shortcut (Windows) or symbolic link (Unix/Linux) to the server's autodeploy directory, and just copy the Web app onto the shortcut each time. For example:
    - Grab tomcat_install_dir/webapps with your right mouse, drag it into devel_dir, release, and choose "Create Shortcut Here".
    - Create struts-test or other applications as subdirectories within devel_dir, as described above.
    - Deploy by using the right mouse to drag struts-test onto the shortcut from step 1, releasing, and choosing "Copy".

Accessing Struts Documentation

- **Read a local copy**
  - This is fastest, but the documentation can get a bit out of date. To read a local copy, install the struts-documentation.war Web app. Copy *struts_install_dir*/webapps/struts-documentation.war to *tomcat_install_dir*/webapps, restart server, and use the URL **http://localhost/struts-documentation/**.
- **Read it from the Apache site**
  - This option is slower, but guarantees that you get the latest versions of the documentation. Use **http://struts.apache.org/learning.html**.
  - The documentation includes FAQs, user guides, tutorials, and the API in Javadoc format.

Adding Struts to an Existing Web Application

- **Copy files into your Web app**
  - Copy JAR files from struts-blank/WEB-INF/lib to your_web_app/WEB-INF/lib.
  - Copy TLD files from struts-blank/WEB-INF to your_web_app/WEB-INF.
  - Copy struts-config.xml from struts-blank/WEB-INF to your_web_app/WEB-INF.
  - Copy the application properties file from struts-blank/WEB-INF/classes/resources to your_web_app/WEB-INF/classes/resources.
  - If you plan on using the automatic validator, copy validation.xml and validator-rules.xml from struts-blank/WEB-INF to your_web_app/WEB-INF.
  - If you plan on using Tiles, copy struts-tiles.xml from struts-blank/WEB-INF to your_web_app/WEB-INF.
- **Update web.xml**
  - Copy declarations out of struts-blank/WEB-INF/web.xml into your_web_app/WEB-INF/web.xml. Most importantly, copy the servlet and servlet-mapping entries that map *.do to org.apache.struts.action.ActionServlet.

Summary

- **Struts is an MVC framework**
  - Also a collection of utilities and custom tag libraries
- **Struts has significant advantages, but adds significant complexity**
  - Compare the use of Struts to the use of normal MVC (especially with the JSP 2.0 expression language)
  - Also consider other frameworks, especially JavaServer Faces (JSF).
- **Test by deploying struts-blank.war**
- **Develop by copying the struts-blank directory**
  - Don't start from scratch; too many changes needed in your Web app

Questions?