



JSF: Introduction, Installation, and Setup

Originals of Slides and Source Code for Examples:
<http://www.coreservlets.com/JSF-Tutorial/>

Customized J2EE Training: <http://courses.coreservlets.com/>
Servlets, JSP, Struts, JSF/MyFaces, Hibernate, Ajax, GWT, Java 5, Java 6, etc. Ruby/Rails coming soon.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



For live JSF 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

Topics in This Section

- **Understanding JSF**
 - Different views of JSF
 - Comparing JSF to standard servlet/JSP technology
 - Pros *and* cons
 - Comparing JSF to Apache Struts
 - Pros *and* cons
- **Setting Up JSF**
 - Downloading and configuring JSF
 - Apache MyFaces
 - Sun Reference Implementation
 - Testing JSF
 - Setting up JSF applications
 - Accessing JSF documentation

4

J2EE training: <http://courses.coreservlets.com>

What is JSF?

- **A set of Web-based GUI controls and associated handlers?**
 - JSF provides many prebuilt HTML-oriented GUI controls, along with code to handle their events.
- **A device-independent GUI control framework?**
 - JSF can be used to generate graphics in formats other than HTML, using protocols other than HTTP.
- **A better Struts?**
 - Like Apache Struts, JSF can be viewed as an MVC framework for building HTML forms, validating their values, invoking business logic, and displaying results.
- **But which is the proper way to view JSF?**
 - The answer depends on what you are going to use it for, but the 1 & 3 are the most common way of looking at JSF.

5

J2EE training: <http://courses.coreservlets.com>

Advantages of JSF (vs. MVC Using RequestDispatcher)

- **Custom GUI controls**
 - JSF provides a set of APIs and associated custom tags to create HTML forms that have complex interfaces
- **Event handling**
 - JSF makes it easy to designate Java code that is invoked when forms are submitted. The code can respond to particular buttons, changes in particular values, certain user selections, and so on.
- **Managed beans**
 - In JSP, you can use `property="*" with jsp:setProperty to automatically populate a bean based on request parameters. JSF extends this capability and adds in several utilities, all of which serve to greatly simplify request param processing.`
- **Expression Language**
 - JSF provides a concise and powerful language for accessing bean properties and collection elements

6

J2EE training: <http://courses.coreservlets.com>

Advantages of JSF (vs. Standard MVC), Continued

- **Form field conversion and validation**
 - JSF has builtin capabilities for checking that form values are in the required format and for converting from strings to various other data types. 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.
- **Centralized file-based configuration**
 - Rather than hard-coding information into Java programs, many JSF 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 without needing to know about the overall system layout.
- **Consistent approach**
 - JSF encourages consistent use of MVC throughout your application.

7

J2EE training: <http://courses.coreservlets.com>

Disadvantages of JSF (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 JSF, 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 JSF as building your actual system.
- **Worse documentation**
 - Compared to the standard servlet and JSP APIs, JSF has fewer online resources, and many first-time users find the online JSF documentation confusing and poorly organized. MyFaces is particularly bad.

8

Disadvantages of JSF (vs. Standard MVC), Continued

- **Less transparent**
 - With JSF applications, there is a lot more going on behind the scenes than with normal Java-based Web applications. As a result, JSF applications are:
 - Harder to understand
 - Harder to benchmark and optimize
- **Undeveloped tool support**
 - There are many IDEs with strong support for standard servlet and JSP technology. Support for JSF is only beginning to emerge, and final level is yet to be determined.
- **Rigid approach**
 - The flip side of the benefit that JSF encourages a consistent approach to MVC is that JSF makes it difficult to use other approaches.

9

Advantages of JSF (vs. Struts)

- **Custom components**
 - JSF makes it relatively easy to combine complex GUIs into a single manageable component; Struts does not
- **Support for other display technologies**
 - JSF is not limited to HTML and HTTP; Struts is
- **Access to beans by name**
 - JSF lets you assign names to beans, then you refer to them by name in the forms. Struts has a complex process with several levels of indirection where you have to remember which form is the input for which action.
- **Expression language**
 - The JSF expression language is more concise and powerful than the Struts bean:write tag.
 - This is less advantageous if using JSP 2.0 anyhow.

10

J2EE training: <http://courses.coreservlets.com>

Advantages of JSF (vs. Struts), Continued

- **Simpler controller and bean definitions**
 - JSF does not require your controller and bean classes to extend any particular parent class (e.g., Action) or use any particular method (e.g., execute). Struts does.
- **Simpler config file and overall structure**
 - The faces-config.xml file is much easier to use than is the struts-config.xml file. In general, JSF is simpler.
- **More powerful potential tool support**
 - The orientation around GUI controls and their handlers opens possibility of simple to use, drag-and-drop IDEs

11

J2EE training: <http://courses.coreservlets.com>

Disadvantages of JSF (vs. Struts)

- **Established base and industry momentum**
 - Struts has a large core of existing developers and momentum among both developers and IT managers; JSF does not.
 - 4/2007 search at dice.com
 - 2308 jobs matching "struts"
 - 829 jobs matching "jsf" (but some listings use "JavaServer Faces")
- **Support for other display technologies**
 - JSF is not limited to HTML and HTTP; Struts is
 - Hey! Didn't I say this was an *advantage* of JSF?
- **Confusion vs. file names**
 - The actual pages used in JSF end in *.jsp*. But the URLs used end in *.faces* or *.jsf*. This causes many problems; in particular, in JSF:
 - You cannot browse directories and click on links
 - It is hard to protect raw JSP pages from access
 - It is hard to refer to non-faces pages in faces-config.xml
- **Self-submit approach**
 - With Struts, the form (*blah.jsp*) and the handler (*blah.do*) have different URLs; with JSF they are the same.

12

J2EE training: <http://courses.coreservlets.com>

Disadvantages of JSF (vs. Struts), Continued

- **Less current tool support**
 - Struts is supported by many widely used IDEs; JSF is not (yet)
- **No equivalent to Tiles**
 - Struts comes with a powerful page layout facility; JSF does not
 - But you can extract Tiles from Struts and use it with JSF
- **Much weaker automatic validation**
 - Struts comes with validators for email address, credit card numbers, regular expressions, and more. JSF only comes with validators for missing values, length of input, and numbers in a given range.
 - But MyFaces has several powerful validators
- **Lack of client-side validation**
 - Struts supports JavaScript-based form-field validation; JSF does not
 - The Shale validators add client-side validation to JSF
- **Worse installation**
 - JSF does not have equivalent of struts-blank to start with
- **POST only**
 - JSF does not support GET, so you cannot bookmark results pages

13

J2EE training: <http://courses.coreservlets.com>

JSF and Struts: The Future

- **Possibilities**
 - JSF will fail and developers that want a framework will stick with Struts
 - JSF will die
 - Doubtful at this point, but other MVC frameworks have failed
 - JSF will flourish and replace Struts
 - Struts will die
 - JSF will grow moderately, and developers will be split
 - Both Struts and JSF will be widely used frameworks
- **Prediction is difficult**
 - Technical factors are not usually what decide these things
- **Recommendations**
 - Move ongoing Struts projects to JSF: not yet
 - Start real-world JSF projects: yes, but with some caution

14

J2EE training: <http://courses.coreservlets.com>

Installing and Configuring JSF: Summary

- **To run JSF, you need three things:**
 - Certain JAR files in WEB-INF/lib
 - Specific to the implementation.
 - Certain web.xml entries
 - Must match the specific implementation. So, if you change MyFaces versions, you have to change your web.xml
 - A blank faces-config.xml file in WEB-INF
 - Standard. Nothing specific to MyFaces.
- **jsf-blank-myfaces**
 - Blank Web app with all three pieces taken from Apache
 - Downloadable from <http://coreservlets.com/JSF-Tutorial/>
 - If you use Eclipse or another IDE, make a project and copy in the above three pieces from jsf-blank
 - If your editor or IDE can use Web apps directly, just copy and rename jsf-blank each time

15

J2EE training: <http://courses.coreservlets.com>

Installing and Configuring JSF: Details

- **Download the JSF JAR files from one of two places**
 - Get `jsf-blank` from coreservlets.com
 - <http://www.coreservlets.com/JSF-Tutorial/>
 - JAR files taken from the MyFaces distribution
 - `web.xml` and `faces-config.xml` from MyFaces examples
 - Download from <http://myfaces.apache.org/download.html>
 - JAR files in `WEB-INF/lib` (some missing!)
 - No sample `web.xml` or `faces-config.xml`
- **Create a blank Web app or new project**
 - With JAR files in `WEB-INF/lib`
 - With `web.xml` entries
 - With `faces-config.xml` in `WEB-INF`
- **Update your class path**
 - Add `myfaces-api-x.x.x.jar` to class path used by compiler or IDE
 - In `WEB-INF/lib` directory of `jsf-blank`

16

J2EE training: <http://courses.coreservlets.com>

Copying Web Applications in MyEclipse: Motivation

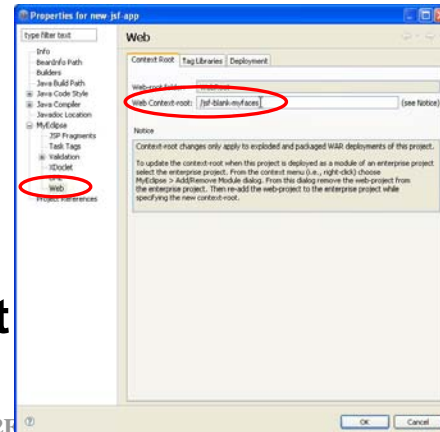
- **Why copy?**
 - Creating new JSF apps in Eclipse or MyEclipse normally results in out-of-date JSF JAR files and settings
 - `jsf-blank-myfaces` is an up-to-date empty app
- **Issue**
 - Cutting/pasting Web app in MyEclipse does not change the context path (i.e., Web application prefix)
 - For example, if you copy `jsf-blank-myfaces` and name the copy `new-jsf-app`, when you deploy, it will still be `http://hostname/jsf-blank-myfaces/...`, not `http://hostname/new-jsf-app/...`
- **Solution**
 - R-click on new project, select Properties, go to MyEclipse → Web, and change Web context-root

17

J2EE training: <http://courses.coreservlets.com>

Copying Web Applications in MyEclipse: Process

1. R-click on jsf-blank-myfaces
2. Select Copy
3. R-click in Web app window
4. Select Paste
5. Rename app
 - E.g., to new-jsf-app
6. R-click on new app
7. Select Properties
8. MyEclipse → Web
9. Change Web context-root



18

Installing and Configuring JSF: Sun Reference Implementation (RI)

- **Download the JSF JAR files from one of two places**
 - Get jsf-blank-Sun-RI from coreservlets.com
 - <http://www.coreservlets.com/JSF-Tutorial/>
 - JAR files taken from the Sun distribution
 - <http://java.sun.com/j2ee/javaserverfaces/download.html>
- **Unzip into a directory of your choice**
 - E.g., on Windows, unzip into C:\Servlets+JSP, resulting in C:\Servlets+JSP\jsf-1_1.
 - I'll refer to this as *install_dir*.
- **Update your class path**
 - Add jsf-api.jar to the class path used by your compiler or IDE
 - In the lib directory of jsf-blank or the jsf_1.1 ZIP file

19

Bookmark the JSF Documentation

- **API Javadocs**
 - <http://java.sun.com/j2ee/javaserverfaces/1.1/docs/api/>
 - Same as the MyFaces version
- **Tag library docs**
 - <http://java.sun.com/j2ee/javaserverfaces/1.1/docs/tlddocs/>
 - Much better than the MyFaces version
- **faces-config.xml annotated DTD**
 - <http://horstmann.com/corejsf/faces-config.html>
- **MyFaces References**
 - <http://myfaces.apache.org/>
 - User's Guide, extensions documentation, FAQs, etc.

Developing a JSF App

- **Start with a sample app containing all necessary entries**
 - Need
 - JAR files
 - web.xml entries
 - config files
 - properties files
 - Neither MyFaces nor Sun RI comes with ready-to-extend blank Web app akin to the Struts struts-blank app
 - Use [jsf-blank](http://coreservlets.com/JSF-Tutorial/) from coreservlets.com/JSF-Tutorial/
- **Use ant**
 - MyFaces source distribution comes with build scripts
 - Not included in main MyFaces download or sample applications
 - Sun RI comes with *install_dir/samples/build.properties.sample*
 - You still need to add web.xml entries

The jsf-blank Web App

- **Ready-to-extend starting point for JSF apps**

- Download from <http://www.coreservlets.com/JSF-Tutorial/>

- **jsf-blank-myfaces**

- Contains all required entries
- Also JAR files for Tomahawk (MyFaces extensions), file upload, and Tiles

- **jsf-blank-myfaces-minimal**

- Contains all required entries
- Does not include any MyFaces extensions

- **jsf-blank-Sun-RI**

- Contains all required entries
- Does not include any extensions



22

Using jsf-blank as Starting Point for New JSF Applications

- **Copy and rename jsf-blank**

- E.g., copy to my-devel-dir as my-first-jsf-app
- Or make a new Eclipse project and copy three main pieces (JAR files, web.xml, faces-config.xml)

- **Add JSF content**

- Edit my-first-jsf-app/WEB-INF/faces-config.xml
- Add JSP, HTML, and other Web content to my-first-jsf-app or my-first-jsf-app/subdirectory
- Add .class files to my-first-jsf-app/WEB-INF/classes

- **Copy app to server's deployment directory**

- E.g., with Tomcat, copy my-first-jsf-app to tomcat-dir/webapps
 - If you do not already have a Java-enabled server set up, see <http://www.coreservlets.com/Apache-Tomcat-Tutorial/>

23

MyFaces Version of jsf-blank (At www.coreservlets.com)

- **Uses the standard .faces extension**
 - Most bundled MyFaces examples use .jsf
 - I prefer .jsf, but .faces is used in the official JSF docs, RI, and all books. Changeable with simple web.xml setting.
- **Uses faces-config.xml**
 - Most MyFaces examples use example-config.xml, but official JSF docs, RI, and all books use faces-config.
 - Again, can be changed with a simple web.xml setting
- **Has all required JAR files in WEB-INF/lib**
 - Taken from three places
 - Main MyFaces download: main jsf API and MyFaces extensions
 - MyFaces sample app: Jakarta commons supporting libraries
 - Various other MyFaces examples: standard.jar (needed for I18N support despite MyFaces docs that say only needed if you use JSTL)
- **Minimal version (no extensions)**
 - Uses only required JAR files

24

J2EE training: <http://courses.coreservlets.com>

JSF Books (In Order of My Personal Preference)

- ***JavaServer Faces, the Complete Reference***
 - By Schalk and Burns
- ***JavaServer Faces in Action***
 - By Kito Mann
- ***Core JavaServer Faces***
 - By Geary and Horstmann
 - Second edition available as of May '07
- ***JavaServer Faces***
 - By Hans Bergsten
- ***Mastering JavaServer Faces***
 - By Dudley, et al

25

J2EE training: <http://courses.coreservlets.com>

Summary

- **JSF provides many useful features for developing complex GUIs & handling events**
- **Standard servlet and JSP technology is a viable alternative**
 - Especially when the MVC approach and the JSP 2.0 expression language is used
- **Struts is also a viable alternative**
 - Future of JSF vs. Struts is unknown
- **To get started**
 - Download jsf-blank from coreservlets.com/JSF-Tutorial
 - MyFaces or Sun-RI versions
 - Set your class path
 - `myfaces-api-x.x.x.jar` (MyFaces) or `jsf-api.jar` (Sun-RI)

26

© 2007 Marty Hall



Questions?

Customized J2EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, Struts, JSF/MyFaces, Hibernate, Ajax, GWT, Java 5, Java 6, etc. Ruby/Rails coming soon.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.