

Useful links

- [J2EE 1.4 tutorial](#)
- [JSF 1.1 release notes](#)
- [java.sun.com homepage for JSF](#)
- [directory of JSF info](#)
- [dmoz.org directory of JSF info](#)
- [JSF spec](#)
- [JSF central](#)

Intro Tutorials

- [JavaRanch JSF tutorial](#)

THINGS TO DO:

- customize the error messages given by the provided validators
- create custom validator that validates multiple fields (like a 3 field date validator)
- add a <http://java.sun.com/j2ee/1.4/docs/tutorial/doc/JSFDevelop5.html> value change event listener . I don't know what they're for yet, but they sound interesting.
- dynamically build the values given by drop-downs (for instance, using info from a database)
- dynamically build a page like the userpages multi-edit page that has the same form repeating an arbitrary number of times.
- create my own custom converter
- do user verification on every page view (essentially verify that the Beacon cookie is valid and corresponds to a valid user in the DB) and send the user to the login page if they're not logged in.
- Make sure that the h:selectOneMenu validator automatically ensures that submitted values actually are from the expected set. If it doesn't then add a validator that ensures it's from the proper set.
- Have one page that sends a user to different pages based on the values entered (for instance, a radio button that selects destination sites). This will also accomplish the goal of "Faces Request Generates Non-Faces Response".
- Get MyEclipseIDE working with Eclipse 3.0RC1 and see if it's a good substitute for Exadel's JSF Studio.
- Eliminate more junk from the JBoss startup so that it doesn't take 10s on each restart.
- make sure the text converter for the decimal field is actually chopping off the numbers past the 2nd decimal. This may be something that's fixed in JSF 1.1.

THINGS DONE

- Have an action method in the managed bean that is called when the page is submitted so that all fields are validated together (thus allowing validation based on multiple fields together, like require that username and password be different).
- dynamically initialize fields particular values and then start the JSF app (or, to take the title of section 2.1.1 of the spec, "Non-Faces Request Generates Faces Response".
- create my own custom validator and have it use its own error messages
- switch the app server used by Exadel's JSF Studio from Tomcat to JBoss.
- use the provided validators and converters to:
 - require field
 - ensure a number is within a particular range
 - ensure a decimal number fits a particular format
- have form values from a submitted page then be displayed on the result page
- create page that, when submitted, goes to another page

EXAMPLE CODE

To iterate through the list of messages for the current page:

Main - JavaServerFaces

```
FacesContext fc = FacesContext.getCurrentInstance();
Iterator messages = fc.getMessage();
FacesMessage fm = (FacesMessage)messages.next();
```