JavaServer Faces 2.0

Sangeetha S
E-Commerce Research Labs,
Infosys Technologies Ltd

Agenda

- JSF 2.0
  - Overview of New Features
  - Facelets
  - Annotations
  - Composite Components
  - Ajax support

Overview

- Major change in the release
- Part of Java Enterprise Edition 6 Platform
- Simplifies/Streamlines web application development
  - Reusable UI components
  - Well defined and simplified transfer of application data to and from UI
  - Easy state management
  - Simplified event handling model
  - Simplified Custom component creation
- Two important major changes in JSF 2.0
  - Annotations
  - Convention used for navigation

Annotations Support

- Annotations allow to mark POJO as managed beans
- Annotations makes faces-config.xml optional (replacement!)
- Marked using @ManagedBean
- Scope of the bean is also specified using annotations
  - Request - @RequestScoped
  - Session - @SessionScoped
  - Application - @ApplicationScoped
- Annotations are used for
  - Converters
  - Validators
  - Composite Components..

Annotations and Managed Beans

- JSF 1.x

```
<managed-bean>
    <managed-bean-name>userBean</managed-bean-name>
    <managed-bean-class>com.demo.UserBean</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```

- JSF 2.0

```
package com.demo;
//import declarations
@ManagedBean
@SessionScoped
public class UserBean { //code }
```

Annotations and Managed Beans

```
package com.demo;
//imports
@ManagedBean(name="regBean")
@RequestScoped
public class RegisterBean {
        //code
}

Usage….

<h:inputText label="eMailID" id="emailId"
value="#{regBean.email}" size="20" required="true"/>
```

Navigation using Convention

- Navigation in JSF 1.x
  - Specified in faces-config.xml

```
<h:commandLink value="Next" action="question2"/>
```

```
<navigation-rule>
  <from-view-id>/question1.jsp</from-view-id>
  <navigation-case>
    <from-outcome>question2</from-outcome>
    <to-view-id>/question2.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
```

Navigation using Convention

- Navigation in JSF 2.0
  - Two types of Navigation
    - Implicit Navigation
    - Conditional Navigation
  - Implicit Navigation
    - Default navigation model
    - No entry in faces-config.xml
    - Value of ***action*** attribute is treated as to-view-id
    - Greatly simplifies navigation
  - Conditional Navigation
    - Set a pre-condition for navigation
    - Pre-condition mandatory for navigation
    - Pre-condition specified using EL

Implicit Navigation

```
<h:commandLink value="Next" action="question2"/>
```

- The value of the action attribute is treated as the to-view-id.
- The default navigation handler appends a '/' and .xhtml extension, and navigation proceeds from question1.xhtml to question2.xhtml.
- In case a bean method is invoked as part of the action, the String values returned by the method are also treated the same way.

Conditional Navigation

```
<navigation-case>
<from-action>#{admin.buyStock}</from-action>
<if>#{user.admin}</if>
<to-view-id>/order.xhtml</to-view-id>
</navigation-case>
```

Facelets

- Facelets – view technology
- Pages are specified using XHTML, files have .xhtml extension
- Pages are compiled into an abstract syntax component tree - UIComponent hierarchy during runtime
- Benefits
  - Page templating
  - Faster execution
  - Portable across diverse platforms
- Libraries supported
  - JSF HTML
  - JSF Core
  - JSF Facelets
  - JSTL Core
  - JSTL Functions

Facelets

- To use Facelets, parameters to be set in web.xml

```
<context-param>
  <param-name>javax.faces.DEFAULT_SUFFIX</param-name>
  <param-value>.xhtml</param-value>
</context-param>

<context-param>
  <param-name>javax.faces.FACELETS_SKIP_COMMENTS</param-name>
  <param-value>true</param-value>
</context-param>
```

Composite Components

- Simplified using
  - Facelets
  - Resources
- Composite component is a Facelet residing in Resources directory
- Defined using
  - <composite:interface>
  - <composite:attribute>
  - <composite:implementation>

Creating Custom Component in JSF 2.0

- Create a resource library - *comp*
- Create a Facelets markup file in the resource library – *register.xhtml*
- Use a composite component in a Facelets page
- Use the following namespace: *http://java.sun.com/jsf/composite/comp*, where *comp* is the name of the resource library.
- Refer to the component as *comp:register*, where *comp* is the prefix that identifies the above namespace and *register* is the name of the Facelets file in the resource library.

General Structure of a Composite Component

- <!-- XHTML DOCTYPE Declaration -->
  <html xmlns="http://www.w3.org/1999/xhtml"
  xmlns:f="http://java.sun.com/jsf/core"
  xmlns:h="http://java.sun.com/jsf/html"
  xmlns:composite="http://java.sun.com/jsf/composite">

  <!—Interface that defines the user contract of the component-->
  <composite:interface> ... </composite:interface>

  <!-- Implementation that specifies the actual markup-->
  <composite:implementation> ... </composite:implementation>

  </html>

Composite Component

- Composite Component attribute can be accessed using

#{cc.attrs.attr_name}

Support for Ajax

- JSF 2.0 supports Ajax integration
- Two ways to send an Ajax request
  - Using the JavaScript function jsf.ajax.request
  - Using <f:ajax> tag
- <f:ajax> tag
  - Has the following attributes
    - render, execute, event, onevent

```
<h:inputText label="eMail ID" id="emailId"
value="#{userBean.email}" size="20"
required="true"
valueChangeListener="#{userBean.validateEmail}">
  <f:ajax event="keyup" render="emailResult"/>
</h:inputText>
```

Bean Validation

Bean Validation Annotation	Constraint Imposed on the element
@NotNull	Cannot be null
@Min	Must be a number whose value must be higher or equal to the specified minimum
@Max	Must be a number whose value must be lower or equal to the specified maximum
@Size	Must be between specified minimum and maximum limits
@Pattern	Must match the specified Java regular expression

JSF 2.0 Bean Validation

- JSF 2.0, provides built-in support for bean validation
- Managed bean properties are mapped to UIInput components on the form
- The JSF implementation can apply the validation constraints on the component's values and capture any error message as FacesMessage

```
@ManagedBean
@RequestScoped
public class RegisterBean {
@Size(min = 1, message = "Please enter the Email")
@Pattern(regexp = "[a-zA-Z0-9]+@[a-zA-Z0-9]+\\.[a-zA-Z0-
9]+", message = "Email format is invalid.")
private String email;
…..
@Size(min = 1, message = "Please enter a Username")
private String userName;
……
@NotNull(message = "Please enter Date of birth") private
Date dob;
….
}
```

Resources

- More details on JSF 2.0 is available at:
  - http://www.developer.com/features/article.php/3867851/JSF-20-Views-Hello-Facelets-Goodbye-JSP.htm
  - http://www.developer.com/java/web/article.php/3868226/JSF-20-Annotations-New-Navigation-Eliminate-XML-Configuration.htm
  - http://www.developer.com/java/web/article.php/3869281/JSF-20-Creating-Composite-Components.htm
    http://www.developer.com/java/web/article.php/3870686/Inside-JSF-20s-Ajax-and-HTTP-GET-Support.htm
  - http://www.developer.com/features/article.php/52691_3880151_1/JSF-20-Bean-Validation-and-Dependency-Injection.htm

  - Authored by Sangeetha S, KL Nitin, Ananya S

Thank You!