Hello Worldwide Web: Your First JSF in JDeveloper

Peter Koletzke
Technical Director &
Principal Instructor

quovera

Now I Remember!

There are three things
I always forget.
Names, faces, and—
the third I can't remember.

—Italo Svevo (1861–1928)

Survey

- Job responsibilities?
  - DBA, developer
- Languages?
  - PL/SQL
  - Java – JSF
  - C++
  - Other
- Tools?
  - Developer Forms/Reports
  - JDeveloper
  - Other

Agenda

- What is JSF?
- Related files
- Accessing the database

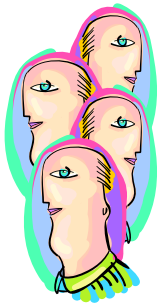
Note: Examples use JDeveloper 11*g* and ADF Faces.

Warning: This material contains more depth than you need to create apps in JDev 11g.

Slides and white paper will be on the NoCOUG website soon.

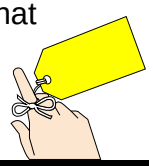
JSF Application

- JavaServer Faces technology
- Relatively new technology
  - Ratified JCP in 5/2004
  - Part of Java EE (v.5)
  - Offers reference implementation – code library
- Effort to simplify JSP development
  - Component-ize it
    - High-level components provide much functionality
  - Integrate the controller
    - No Struts needed
  - Write less HTML (than JSP)
    - Component handles HTML writing
- JSF is often run in a JSP file
  - XML-like tags: elements and attributes



JSF Code

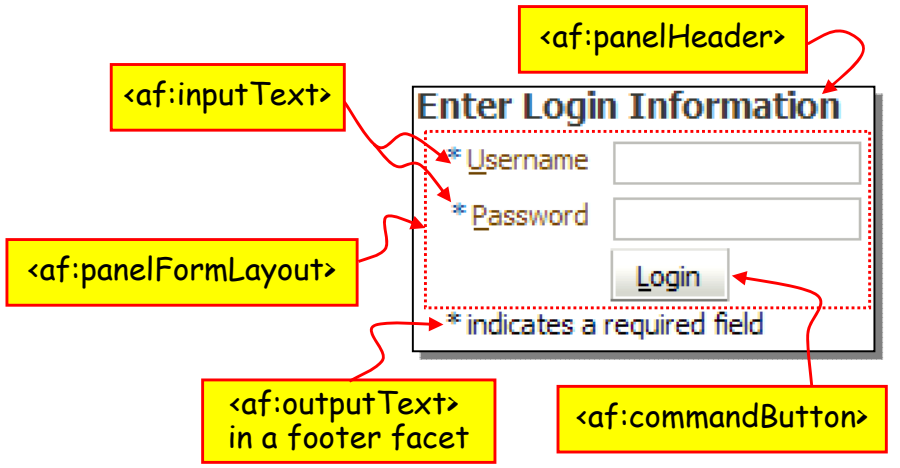
- Usually used for a web-based application
  - Technically can be any client – PDA, cell phone
- XML-like (or just plain old XML)
  - Start and end tag required
  - Elements, commonly called “components”
  - Attributes
- Its components represent JSP *action tags*
  - Requires a prefix and tag library definition (.tld file) to back it
  - Tag library definition points to the Java class that implements the tag
- Following example mixes standard JSF with ADF Faces Rich Client



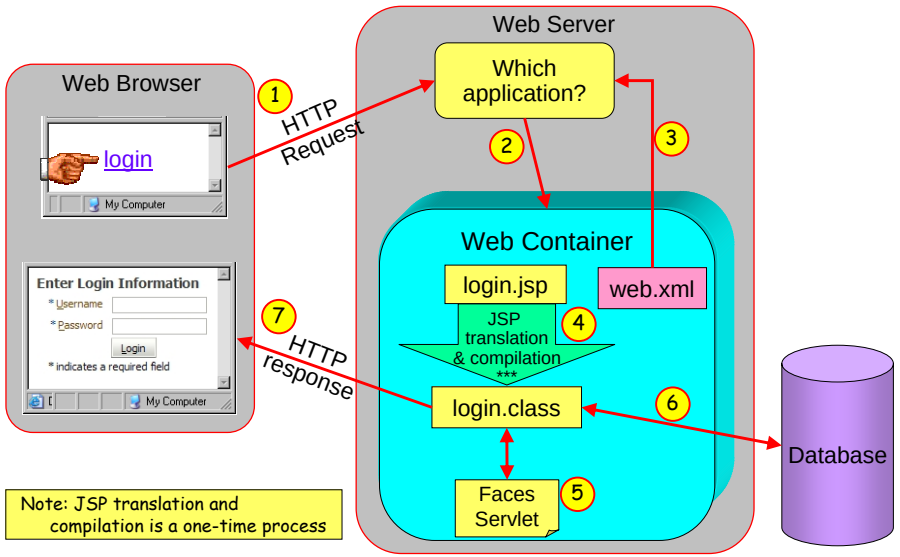
JSF Code Snippet

```
<af:form id="loginForm"
    binding="#{backing_login.loginForm}">
  <af:panelHeader text="Enter Login Information"
       binding="#{backing_login.ph1}" id="ph1">
  <af:panelFormLayout binding="#{backing_login.pfl1}"
       id="pfl1">
    <f:facet name="footer">
      <af:outputText value="* indicates a required field"
          binding="#{backing_login.ot1}" id="ot1"/>
    </f:facet>
...
  <af:inputText binding="#{backing_login.usernameField}"
      id="usernameField" required="true" maximumLength="20"
      columns="15" requiredMessageDetail="is required"
      labelAndAccessKey="&amp;Username" shortDesc="Your name"/>
...
  <af:commandButton binding="#{backing_login.loginButton}"
      id="loginButton" textAndAccessKey="&amp;Login"
      action="#{backing_login.loginButton_action}"/>
```


The Result



JSF Communication Process

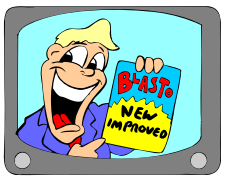


The Steps

1. The browser issues an HTTP request to the web server.
2. The web server determines the application and passes the request to the web container (WLS or OC4J)
3. The web server reads web.xml to determine that this is a JSF JSP.
4. The JSP is translated to a servlet and compiled (the first time it is run)
5. The web server passes the request to the Faces Servlet. The Faces Servlet instantiates a life cycle class, which processes the JSP JSF.
6. The servlet accesses any necessary Model layer code to obtain values from the data source (such as a database).
7. The Faces Servlet then assembles the page and sends it using an HTTP response to the browser.

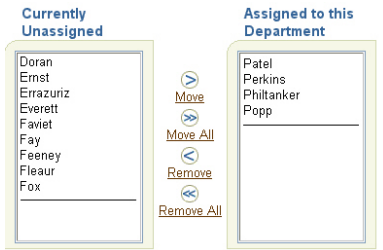
JSF Features

- Rich component set
  - Core library – for application tasks
  - HTML library – for HTML tags, forms
  - JSP tag library included
    - Can be implemented in other languages
  - Include data binding properties
- Embedded controller
  - Previously, no standard controller for JSPs
  - Struts was/is a popular controller framework
- Event-driven
  - Events on the component level
  - Like Forms triggers



ADF Faces

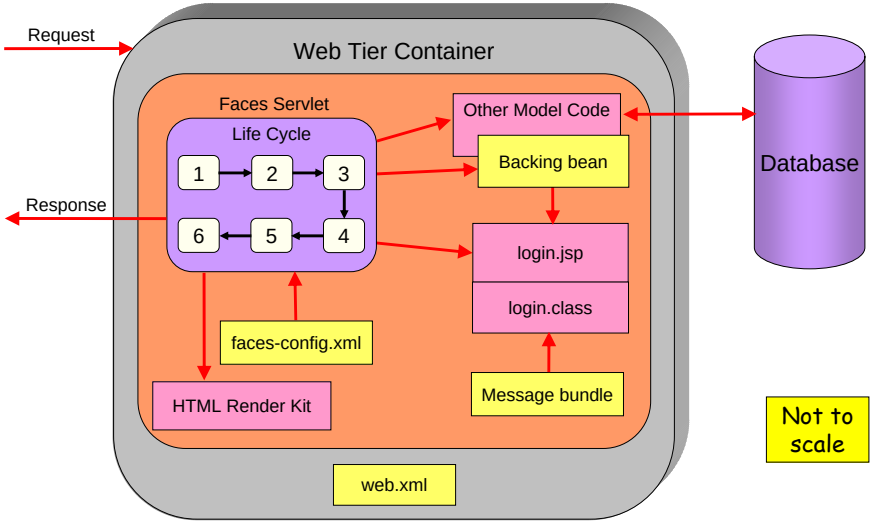
- Oracle JSF component libraries
  - Released to MyFaces open source project in Jan. 2006
    - Trinidad project at myfaces.apache.org
  - Available in JDeveloper 10.1.3 as ADF Faces
  - In JDeveloper 11g as ADF Faces Rich Client
- Implements components available in UIX
  - Uses JSF mechanisms
  - Adds even more functionality to JSF
  - Over 150 components, For example, selectOrderShuttle:



Agenda

- What is JSF?
- Related files
- Accessing the database

JSF Files



The Files

- **web.xml** – used to start FacesServlet, which instantiates and runs the life cycle class
- **faces-config.xml** – the controller file used to manage page flow
- **Backing bean** – code for the components on the page
- **Message bundle** – supplies text for the JSP
- **login.jsp** – JSF (JSP) source code that is compiled into login.class
- **Model layer code** – used to obtain values from the data source (such as a database).
- **HTML render kit** – converts components in memory to HTML tags

web.xml

- web.xml –web module deployment descriptor
  - Contains an entry such as this:

```
<welcome-file-list>
  <welcome-file>forum_query.jsp</welcome-file>
</welcome-file-list>
```

- Contains the *URL pattern* used to determine which servlet will take control
- Contains the servlet name and class file name

web.xml Snippet

```
<servlet>
  <servlet-name>Faces Servlet</servlet-name>
  <servlet-class>javax.faces.webapp.FacesServlet
  </servlet-class>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>/faces/*</url-pattern>
</servlet-mapping>
 <servlet-mapping>
    <servlet-name>resources</servlet-name>
    <url-pattern>/adf/*</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>resources</servlet-name>
    <url-pattern>/afr/*</url-pattern>
  </servlet-mapping>
```


faces-config.xml

- Standard Java EE file
  - The “application configuration resource file”
  - Located in the WEB-INF directory
- The JSF file that defines controller actions
  - Navigation rules
    - Define the “from” page for a page flow
  - Navigation cases
    - Define the “to” page for a page flow
  - Managed bean definitions
  - Render kits
  - Converters and validators

Code View of Navigation Rules

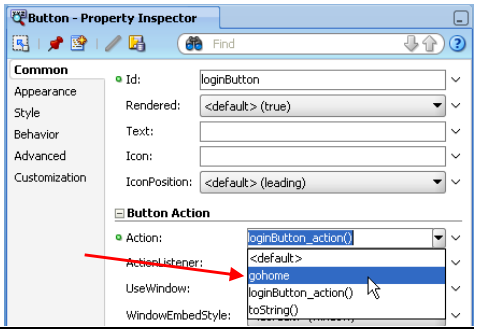
```
</faces-config>
 <navigation-rule>
    <from-view-id>/login.jsp</from-view-id>
    <navigation-case>
      <from-outcome>login</from-outcome>
      <to-view-id>/home.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <from-view-id>/home.jsp</from-view-id>
    <navigation-case>
     <from-outcome>logout</from-outcome>
     <to-view-id>/login.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
</faces-config>
```


Navigation Case Outcome

- In addition to the “to” page, a navigation case is assigned an *outcome*

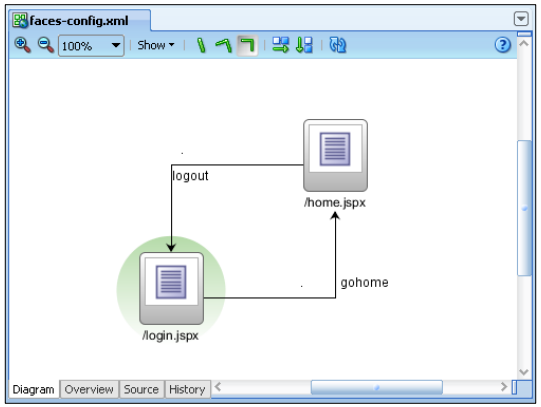
```
<navigation-case>
  <from-outcome>gohome</from-outcome>
  <to-view-id>/home.jsp</to-view-id>
</navigation-case>
```

- Navigation occurs when *action* property of a button is set to the outcome name

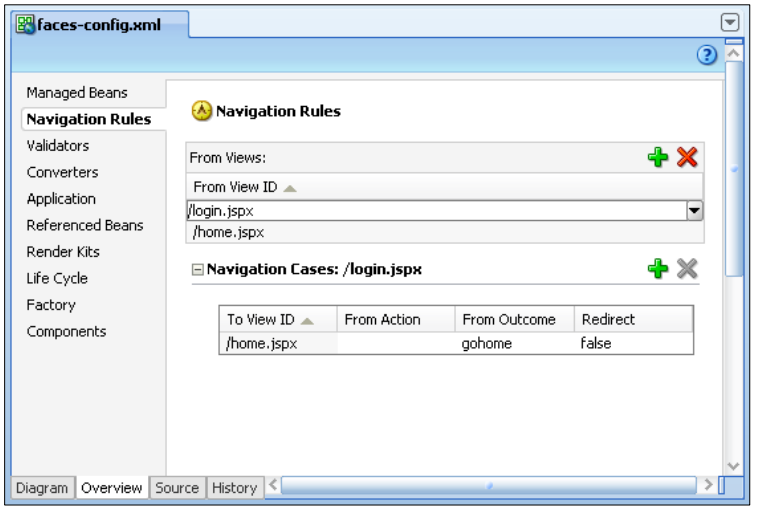


Editing faces-config.xml

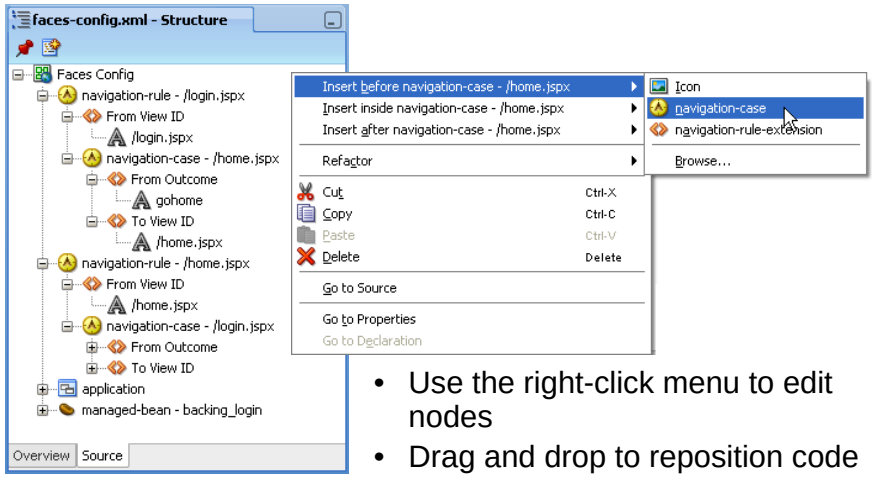
- JSF Navigation Diagram
  - Look under WEB-INF
  - Double click the faces-config.xml file in the navigator
  - Use drag and drop to add elements



Editing in the Overview Tab



Editing in the Structure Window



- Use the right-click menu to edit nodes
- Drag and drop to reposition code
- OR use the code editor

Backing Beans

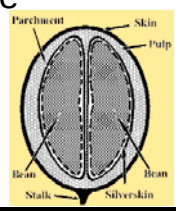
- Backing bean: a Java class file used for code pertaining to a specific page
  - For example, login.jsp would use a Login.java backing bean
- Contain accessors for components on the page and other code for just that page
- Optional file: only create it if you need to change the logic
- These are registered in faces-config.xml:

```
<managed-bean>
  <managed-bean-name>backing_login</managed-bean-name>
  <managed-bean-class>login.view.backing.Login</managed-bean-class>
  <managed-bean-scope>session</managed-bean-scope>
  <!--oracle-jdev-comment:managed-bean-jsp-link:1login.jsp-->
</managed-bean>
```

Comment line declares that JDeveloper will create accessors when you add a component to the JSF page.

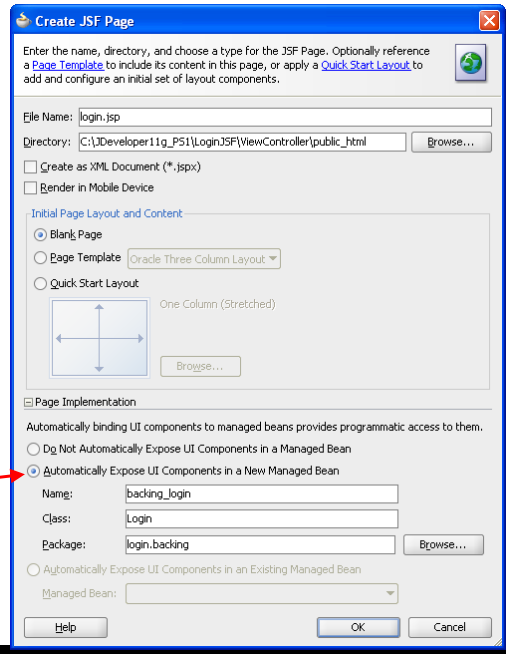
Backing Bean Contents

- Variables for each component
- Setters and Getters for each component
- If JDeveloper created the backing bean it will maintain it
  - Uses the comment line shown earlier in the faces-config.xml file
  - Adding a component adds the variable and accessors for that component
  - Deleting a component removes them
  - Renaming a component renames them



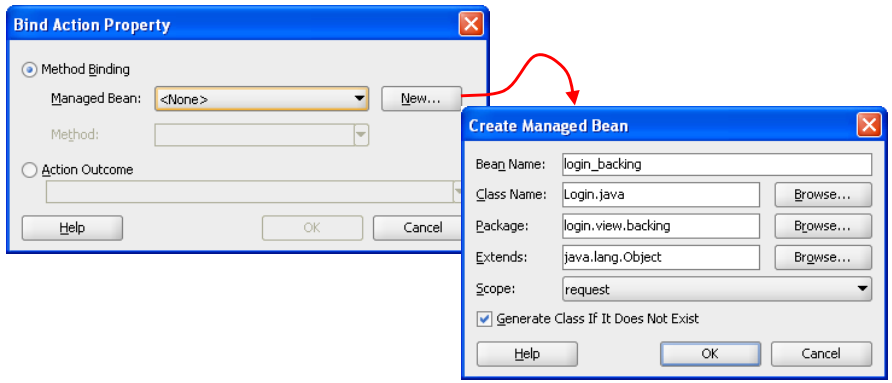
Creating the Bean

- Create a Java class from the New Gallery
  - Enter it in faces-config manually
- OR from the Create JSF Page dialog
  - Specify the name in faces-config and the class file name



Alternative for Creating the Backing Bean

- Double click an action component (button or link)
  - These dialogs will set up the Java class and register it in faces-config



Managed Beans or Backing Beans?

- A *bean* (JavaBean) is a Java class file with a standard set of methods
- *Managed bean* is a Java class file used to handle operations and data for a resource – such as a JSF page
- *Backing bean* is a managed bean that supports a specific page
- The terms “managed bean” and “backing bean” are sometimes used interchangeably

About Scope

- Values in a bean or a FacesContext variable are cleared out of memory at certain times
- You can declare a *scope* for these objects:
  - *request*: for the HTTP request/response
  - *session*: for the user's session with the app server (until they log out or time out)
  - *application*: across client sessions; held in the app server memory
- ADF Controller offers additional scopes
  - *pageFlow*
  - *view*
  - *backingBean*

Backing Bean Snippet

```
package login.view.backing;

import java.util.ResourceBundle;

import javax.faces.application.FacesMessage;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
// more imports

public class Login
{
  private RichForm loginForm;
  private RichDocument d2;
  private RichPanelHeader ph1;
  private RichPanelFormLayout pfl1;
  private RichInputText usernameField;
  private RichInputText passwordField;
  private RichCommandButton loginButton;
  private RichOutputText ot1;
  int loginAttempts = 0;
```

- Imports and private variables for each component

Backing Bean Snippet (continued)

```
  public void setLoginForm(RichForm form)
  {
    this.loginForm = form;
  }

  public RichForm getLoginForm()
  {
    return loginForm;
  }
 public void setUsernameField(RichInputText it1)
  {
    this.usernameField = it1;
  }

  public RichInputText getUsernameField()
  {
    return usernameField;
  }

  // and accessors for all other components
}
```

- Getters and setters for each component
- Other contents: validation code

Message Bundles

- Also called "resource bundles"
- Separate properties (text) or Java class file containing labels and messages
- Linked to the page through expressions on the components
- Also readable by code in the backing bean
- Allow for centralization of messages
- Automated localization and internationalization (language-specifics)

Message Bundles in JDeveloper

- Define the message bundle name
  - Project properties – Resource bundle page
- Add the message using a dialog
  - Select “Select Text Resource” from the pulldown by an applicable component
- Refer to the message using Expression Language, for example:

  `#{viewcontrollerBundle.WELCOME_HOME}`
- This expression will be resolved at runtime and at design time

Message Bundle Snippet

```
#English welcome message
WELCOME_HOME=Welcome Home

# login attempt messages
incorrectLogin=Incorrect login. Try again.
loginHint=You seem to have forgotten the password.
```

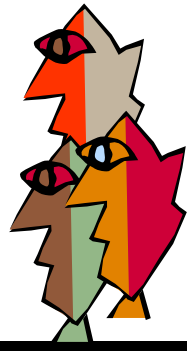
Demo 3

Agenda

- What is JSF?
- Related files
- Accessing the database

The JDeveloper Technique

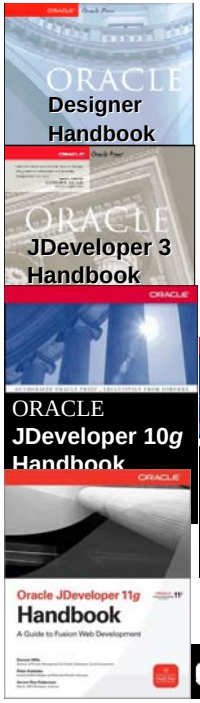
- Create ADF Business Components (ADF BC)
  - Declares database structures (tables and views)
  - Similar functionality to EJBs
- Drop data controls onto the page
  - This binds the components to the ADF BC components
  - A separate PageDef file is created for the bindings



Summary

- JSF evolved to make web development easier
- Some awareness of the runtime environment and life cycle will help in your first JSF
- You need to create the JSF JSP file
- You also need supporting files:
  - web.xml – created automatically to assist in loading pages
  - faces-config.xml – the main JSF configuration file
  - Backing beans – programmatic code for the page
  - Message bundles – centralized text strings for the page
- JDeveloper offers many tools to assist
  - Including frameworks to access the database

What a surprise!



- Please fill out the evals
- Books co-authored with Dr. Paul Dorsey, Avrom Roy-Faderman, & Duncan Mills
- Personal web site:
  `http://ourworld.compuserve.com/homepages/Peter_Koletzke`

quovera

http://www.quovera.com

- Founded in 1995 as Millennia Vision Corp.
- Profitable for 7+ years without outside funding
- Consultants each have 10+ years industry experience
- Strong High-Tech industry background
- 200+ clients/300+ projects
- JDeveloper Partner
- More technical white papers and presentations on the web site