Moving JavaServer Faces Forward
JSF 2.2 Update
Ed Burns
@edburns
www.java.net/blogs/edburns
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Plug: See Oracle’s JavaEE Cloud Story

- Jürgen Höller is giving Spring’s cloud story right now in the Birnhorn room
- I will be giving Oracle’s JavaEE cloud story at 14:00 today in this room.
Program Agenda

- Where is JSF in its lifecycle?
  - Summarize the value proposition JSF has been trying to establish
  - Evaluate to what extent JSF has delivered on that value proposition
- What’s in JSF 2.2?
  - Big ticket features
  - Smaller features
  - Current status
- Questions and Answers
Program Agenda

• Where is JSF in its lifecycle?
  • Summarize the value proposition JSF has been trying to establish
  • Evaluate to what extent JSF has delivered on that value proposition
• What’s in JSF 2.2?
  • Big ticket features
  • Smaller features
  • Current status
• Questions and Answers
Where is JSF in its lifecycle?

- JavaOne 2001 Technical Keynote

**JavaServer Faces: Definition**

- GUI Component Architecture for JavaServer applications
  - GUI runs on server, renders to client
  - Client posts events back to server
Where is JSF in its lifecycle?

• JavaOne 2001 Technical Keynote

JavaServer Faces: Motivation

• Eliminate burden on developers
  – Today must create/maintain own frameworks
• Boost Tools, 3rd Party Component support
  – easier to leverage single framework
• Improve GUI quality
  – tools & framework do the hard stuff
Where is JSF in its lifecycle?

• JavaOne 2001 Technical Keynote

JavaServer Faces: Design Focus

• Tools, tools, tools!
• Lightweight GUI base classes to address state & lifecycle issues
• Event model for connecting client-side events to server-side app behavior
• Input validation
• Internationalization, Localization, Accessibility
• Renders markup appropriate for target client
Where is JSF in its lifecycle?

- Original JSF Value Proposition
  - Solid web framework foundation
  - Vibrant 3rd party component market
  - Widely available in JavaEE containers
  - Open development process
Where is JSF in its lifecycle?

- Original JSF Value Proposition
- Solid web framework foundation
  - Vibrant 3rd party component market
  - Widely available in JavaEE containers
  - Open development process
Where is JSF in its lifecycle?

• Original JSF Value Proposition
✓ • Solid web framework foundation
✓ • Vibrant 3rd party component market
   • Widely available in JavaEE containers
   • Open development process
Where is JSF in its lifecycle?

- Original JSF Value Proposition
- Solid web framework foundation
- Vibrant 3rd party component market
- Widely available in JavaEE containers
  - Open development process
Where is JSF in its lifecycle?

- Original JSF Value Proposition
- Solid web framework foundation
- Vibrant 3rd party component market
- Widely available in JavaEE containers
- Open development process
Where is JSF in its lifecycle?

- Original JSF Value Proposition
- Solid web framework foundation
- Vibrant 3rd party component market
- Widely available in JavaEE containers
- Open development process

Future Prognosis

- Continued support and investment from Oracle via JCP and JavaEE container team
- Staying true to its original mission
  - I think it would be a mistake to push JSF straight into the thin server model
  - There still is a place for UI state mostly on the server, perhaps with a mini-lifecycle in the client
Where is JSF in its lifecycle?

**JSF Timeline**

- JSR 127
  - JSF 1.0 11 March 2004
  - JSF 1.1 27 May 2004
  - JSR 252
  - JSF 1.2 11 May 2006
  - JSF 1.2 Maintenance Release 1 19 December 2006
  - JSF 1.2 Maintenance Release 2 13 June 2008
  - JSF 1.2 Maintenance Release 3 25 August 2008
- JSR 314
  - JSF 2.0 1 July 2009
  - JSF 2.1 16 July 2010
  - JSF 2.1 Maintenance Release 2 22 November 2010
- JSR 344
  - Started 14 April 2011
  - Early Draft Review released 8 December 2011
  - Working toward Public Review Draft
Program Agenda

- Where is JSF in its lifecycle?
  - Summarize the value proposition JSF has been trying to establish
  - Evaluate to what extent JSF has delivered on that value proposition
- What’s in JSF 2.2?
  - Big ticket features
  - Smaller features
  - Current status
- Questions and Answers
Program Agenda

• Where is JSF in its lifecycle?
  • Summarize the value proposition JSF has been trying to establish
  • Evaluate to what extent JSF has delivered on that value proposition

• What’s in JSF 2.2?
  • Big ticket features
  • Smaller features
  • Current status

• Questions and Answers
What’s new in JSF 2.2?

Big Ticket Features

- Faces Flows
- Multi Templating
- HTML 5
What’s new in JSF 2.2?

Big Ticket Features: Faces Flows

• Lack of formal way to group pages leads to maintenance problems
  • Hard to understand navigation
  • Like a large program that is a single function with lots of goto statements
• Faces Flows goals
  • Allow grouping related views together
  • Allow building applications as composites of modules of functionality
What’s new in JSF 2.2?
Big Ticket Features: Faces Flows

• JSF Navigation Story Recap
• Defines a graph of VDL views
• XML rule base defines relationship between views
  • Implicit rules added in JSF 2.0

<navigation-rule>
  <description>
    global navigation rules
  </description>
  <from-view-id>*</from-view-id>
  <navigation-case>
    <description>Go to Edit Project Page</description>
    <from-outcome>editProject</from-outcome>
    <to-view-id>/project/edit.jsf</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Select Projects Page</description>
    <from-outcome>showProjects</from-outcome>
    <to-view-id>/project/show.jsf</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Edit Sprint Page</description>
    <from-outcome>editSprint</from-outcome>
    <to-view-id>/sprint/edit.jsf</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Dashboard</description>
    <from-outcome>showDashboard</from-outcome>
    <to-view-id>/dashboard/show.jsf</to-view-id>
  </navigation-case>
</navigation-rule>

<navigation-rule>
  <description>
    dashboard navigation rules
  </description>
  <from-view-id>/dashboard/show.xhtml</from-view-id>
  <navigation-case>
    <description>Go to Edit Project Page</description>
    <from-outcome>editProject</from-outcome>
    <to-view-id>/project/edit.jsf</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Select Projects Page</description>
    <from-outcome>showProjects</from-outcome>
    <to-view-id>/project/show.jsf</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Edit Sprint Page</description>
    <from-outcome>editSprint</from-outcome>
    <to-view-id>/sprint/edit.jsf</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Dashboard</description>
    <from-outcome>showDashboard</from-outcome>
    <to-view-id>/dashboard/show.jsf</to-view-id>
  </navigation-case>
</navigation-rule>

<navigation-rule>
  <description>
    project navigation rules
  </description>
  <from-view-id>/project/show.xhtml</from-view-id>
  <navigation-case>
    <description>Go to Show Sprints</description>
    <from-outcome>showSprints</from-outcome>
    <to-view-id>/sprint/show.xhtml</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Show Stories</description>
    <from-outcome>showStories</from-outcome>
    <to-view-id>/story/show.xhtml</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Show Dashboard</description>
    <from-outcome>showDashboard</from-outcome>
    <to-view-id>/dashboard/show.xhtml</to-view-id>
  </navigation-case>
</navigation-rule>

<navigation-rule>
  <description>
    sprint navigation rules
  </description>
  <from-view-id>/sprint/show.xhtml</from-view-id>
  <navigation-case>
    <description>Go to Show Stories</description>
    <from-outcome>showStories</from-outcome>
    <to-view-id>/story/show.xhtml</to-view-id>
  </navigation-case>
  <navigation-case>
    <description>Go to Show Dashboard</description>
    <from-outcome>showDashboard</from-outcome>
    <to-view-id>/dashboard/show.xhtml</to-view-id>
  </navigation-case>
</navigation-rule>
What’s new in JSF 2.2?

Big Ticket Features: Faces Flows

- Faces Flows builds on the existing JSF navigation story and adds
  - Several node types in addition to VDL views
  - Well defined start and end-node concept
  - Scope concept
  - Ability to package a flow as a module and enclose it in a self-configuring jar
- Still described in XML
  - In faces-config.xml file(s)
  - In VDL views, within <f:metadata>
- Two main varieties of flows
  - Unbounded (No specific start or end, not within a specific flow)
  - Bounded (with explicit start node and end node(s))
What’s new in JSF 2.2?

Big Ticket Features: Faces Flows

- Bounded and Unbounded Faces Flows
What’s new in JSF 2.2?
Big Ticket Features: Faces Flows

• Runtime API: javax.faces.flow package
What’s new in JSF 2.2?
Big Ticket Features: Faces Flows

• Runtime API
  • @FlowScoped CDI annotation
  • #{facesFlowScope} EL implicit Object
What’s new in JSF 2.2?
Big Ticket Features: Faces Flows

- Demo
What’s new in JSF 2.2?

Big Ticket Features

• Faces Flows
• Multi Templating
• HTML 5
What’s new in JSF 2.2?

Big Ticket Features: Multi-templating

• Build on existing concepts in Facelets
• Enables defining a contract for a “site template”, similar to Joomla!
• Contract specifies
  • Template metadata: name, version, author, etc.
  • Facelet insertion points: header, menu, content, etc.
• CSS classes
• Templates can be bundled in self configuring jar and dropped into WEB-INF/lib to activate.
What’s new in JSF 2.2?
Big Ticket Features: Multi-templating

- Demo
What’s new in JSF 2.2?

Big Ticket Features

- Faces Flows
- Multi Templating
- HTML 5
What’s new in JSF 2.2?
Big Ticket Features: HTML5

• What’s the big deal with HTML5?
  • It lets the UI authoring experience focus on the browser, instead of server-side processing that delivers content to the browser.
• Recall my JSF Days 2008 keynote

What Makes A Distributed App?

Finding the best allocation of processing tasks to processing nodes

• User Interface
• Domain Logic
• Application Logic
• Data Persistence
• Communication
• Reliability, Security
What’s new in JSF 2.2?
Big Ticket Features: HTML5

• JSF component libraries already can support HTML5
• For JSF 2.2, the goal is to exploit new concepts in HTML5 where sensible
• HTML5 Content Categories
  • Metadata
    • New features in base, command, link, meta, noscript, style, title
  • Sectioning and Heading
    • New elements nav, aside, article, section, h1-h6, hgroup
  • Form associated elements
    • New features in button, input, object, textarea, label
    • New elements fieldset, keygen, meter, output, progress, select
  • Input related elements
    • New elements autocomplete, autofocus, pattern, placeholder, type
  • data-xxx attributes
Program Agenda

• Where is JSF in its lifecycle?
  • Summarize the value proposition JSF has been trying to establish
  • Evaluate to what extent JSF has delivered on that value proposition
• What’s in JSF 2.2?
  • Big ticket features
  • Smaller features
  • Current status
• Questions and Answers
What’s new in JSF 2.2?
Smaller Features: <f:viewAction>

• Directly taken from Seam’s Faces Module <s:viewAction>
• Think of it like a button that automatically clicks itself when the page loads

<f:metadata>
    <f:viewParam id="id" name="id" value="#{newsReader.selectedStoryId}"
        required="true"
        requiredMessage="You did not specify a headline. (The id parameter is missing)"
        converterMessage="Invalid headline. (The id parameter is not a number)"
        validatorMessage="Invalid headline. (The id parameter is not a positive number)">
        <f:validateLongRange minimum="1"/>
    </f:viewParam>
    <f:viewAction action="#{newsReader.goToPage01IfValidationFailed}"
        phase="PROCESS_VALIDATIONS" onPostback="true" />
    <f:viewAction action="#{newsReader.loadStory}" onPostback="true" />
</f:metadata>
What’s new in JSF 2.2?
Smaller Features: CSRF Protection

• What is Cross Site Request Forgery (CSRF, pronounced SEE-surf)?
  • Trick the browser into sending requests that the user did not actually intend to initiate
What’s new in JSF 2.2?
Smaller Features: CSRF Protection

• How does JSF protect your app against this attack?
  1. It already does and always has!
     POSTback is a virtue. JSF 2.2 just makes encryption of the view state on by default
What’s new in JSF 2.2?
Smaller Features: CSRF Protection

• How does JSF protect your app against this attack?
  2. For non-POSTback requests, the <protected-views> element comes to the rescue.
    • ViewHandler asks “Is this view protected?” if so, proceed.
    • If not, inspect the Referer[sic] header. If coming from a protected view, proceed.
    • If not, does the referrer at least originate in this web app? If so, proceed.
    • If no Referer header, look for
      ResponseStateManager.NON_POSTBACK_VIEW_TOKEN_PARAM
      This query parameter will be present because of these changes to
      ResponseStateManager.getState()
What’s new in JSF 2.2?
Smaller Features: More CDI injectable artifacts

<table>
<thead>
<tr>
<th>Artifact Type</th>
<th>Injected if declared in application configuration resources (faces-config.xml)</th>
<th>Injected if declared via Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>javax.faces.event.ActionListener</td>
<td>yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>javax.el.ELResolver</td>
<td>yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>javax.faces.application.NavigationHandler</td>
<td>yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>javax.faces.application.ResourceHandler</td>
<td>yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>javax.faces.applicationStateManager</td>
<td>yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>javax.faces.event.SystemEventListener</td>
<td>yes</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>javax.faces.component.UIComponent</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>javax.faces.component.behavior.Behavior</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>javax.faces.convert.Converter</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>javax.faces.validator.Validator</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
What’s new in JSF 2.2?
Smaller Features: FaceletFactory now in the API

- Yes, it is now programmatically possible to instantiate a Facelet and call `apply()` on it.
- Why would you need to do this?
- Because the result of `Application.createComponent()` is not always enough
  - No automatic ajaxification
  - No child handlers
  - Other cases
What’s new in JSF 2.2?
Smaller Features: `<h:inputFile>`

- No need for commons fileupload
- Ajax and non-Ajax based
- Portlet safe
What’s new in JSF 2.2?
Smaller Features: ClientWindow

• New class `javax.faces.lifecycle.ClientWindow`
• Represents a browser tab, browser window, pop-up, portlet, or anything else that can display a UIComponent hierarchy rooted at a UIViewRoot
• Not-enabled by default because it can be invasive in the URL space
  • “none” mode: the feature is disabled
  • “url” mode: the ClientWindow is encoded in the URL
• Faces Flows is entirely dependent on this feature
Program Agenda

• Where is JSF in its lifecycle?
  • Summarize the value proposition JSF has been trying to establish
  • Evaluate to what extent JSF has delivered on that value proposition
• What’s in JSF 2.2?
  • Big ticket features
  • Smaller features
  • Current status
• Questions and Answers
Current Status

- Big Ticket Features still in progress
  - Good progress on Faces Flows just this week!
  - Multi-templating has an existing prototype
  - HTML5 work is straightforward
- Smaller items all implemented with a few exceptions
  - File upload needs Ajax support
  - ClientWindow needs some additional polish
  - Relative ResourceHandler not implemented
  - Better integration with j_security_check and JSF
  - Runtime inspection and modification of config process
Program Agenda

- Where is JSF in its lifecycle?
  - Summarize the value proposition JSF has been trying to establish
  - Evaluate to what extent JSF has delivered on that value proposition
- What’s in JSF 2.2?
  - Big ticket features
  - Smaller features
  - Current status
- Questions and Answers
Questions and Answers
Safe Harbor Statement

The preceding was intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.