Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

Writing Killer JavaServer™ Faces 2.0 Components

Kito D. Mann
Principal Consultant
Virtua

Kito D. Mann

- Principal Consultant at Virtua
  - Training, consulting, architecture, mentoring, JSF product development
- Author, JavaServer Faces in Action
- Founder, JSF Central
  - http://www.jsfcentral.com
- Internationally recognized speaker
  - JavaOne, JavaZone, Javapolis, NFJS, etc.

Kito D. Mann

- JCP Member
  - JSF 2.0, WebBeans, JSF Portlet Bridge, Design-Time Metadata for JavaServer Faces Components, Portlets 2.0

Agenda

- Writing a Simple Compsoite Component
- Adding Custom Logic
- Working with Resources
- Using the Ajax API
- Summary

Writing a Simple Composite Component

- Single View Definition Language (VDL) file
- Contains markup and text
- Can contains other JSF components
- Similar to Facelet compositions

DEMO

A Simple JSF Component

Adding Custom Logic

- If you need custom logic, you must write a class
  - Decode behavior
  - Specific properties
  - Event listeners
- Class can be written in script
- If necessary, you can still write a separate Renderer class

DEMO

Adding Custom Logic

Resource Handling

- *Resources* are items other than JSF views
  - Images, style sheets, JavaScript files, etc.
  - Can be versioned
- *Libraries* contain multiple resources
  - Can be versioned and localized

Resource Handling

- Resources can be loaded from:
  - Web application root:

    `resources/<resourceIdentifier>`
  - JAR file:

    `META-INF/resources/<resourceIdentifier>`
- VDL components placed inside libraries

Referencing Resources

- Referenced via the VDL:
  - <h:graphicImage name=”Planets.gif” library=”images”/>
  - <h:graphicImage value=”#{resource[‘images:Planets.gif’]}”/>
  - <h:outputScript name=”ajax.js” library=”javax.faces” target=”head”/>
- Expression can be used inside of CSS and JavaScript files

Referencing Resources

- @ResourceDependency annotation for component or renderer implementations

DEMO

Working with Resources

Using the Ajax API

- All methods under `javax.faces.Ajax` namespace
- Collecting and encoding view state:

```
var options = {inputs:'id1', 'id2', 'id3'};
var viewState =
javax.faces.Ajax.viewState(form, options);
```

Using the Ajax API

- Partial tree traversal:
  - javax.faces.Ajax.ajaxRequest
- Partial page update
  - javax.faces.Ajax.processAjaxResponse
- Shortcut: <f:ajax> tag

DEMO

Using the Ajax API

Summary

- Writing components in JSF 2.0 is easy
  - Single VDL page
  - Optional Java or Script class
  - Resource/library support for JavaScript, CSS, etc.
  - Integrated Ajax support

Resources

- JSR 314 home page
  - http://www.jcp.org/en/jsr/detail?id=314
- Public JSF 2 java.net project
  - https://javaserverfaces-spec-public.dev.java.net/
- Mojarra (JSF 2 RI)
  - https://javaserverfaces.dev.java.net/
- JSF 2 Group blog
  - http://blogs.jsfcentral.com/jsf2group
- Ryan Lubke's blog
  - http://blogs.sun.com/rlubke/
- Ed Burns' blog
  - http://weblogs.java.net/blog/edburns/

Questions