JavaOne
Sun's 2004 Worldwide Java Developer Conference

Putting Faces on Your Portlets

Exploiting JavaServer™ Faces Technology in Portlet Applications

Brendan Murray

Senior Software Architect
IBM Dublin Software Lab

http://www.ibm.com

Why Are We Here?

To Learn …

… what it means to create and run
a Portlet that uses JavaServer™ Faces
technology

Agenda

- Introduction
- JavaServer™ Faces technology and Portlets
- Creating your JSF Portlet
- Demo

Agenda

- Introduction
    - What we’ll cover today
    - What is JavaServer™ Faces technology?
    - What are Portlets?
- JavaServer™ Faces Technology and Portlets
- Creating your JSF Portlet
- Demo

Introduction

What We'll Cover Today

- Infrastructure and basic portlet support in JSF
- Implementation and customization of this support
- Creation and deployment of a simple portlet

Introduction

What Is JavaServer™ Faces Technology?

- JCP JSR 127
- Spec leads: Ed Burns and Craig McClanahan, Sun Microsystems
- Simplifies building Java™ 2 Platform, Enterprise Edition (J2EE™) technology-based user interfaces
  - Page contains reusable components
  - Client-side events wired to server events

Introduction

What Are Portlets?

- JCP JSR 168
- Spec leads: Steffan Hepper, IBM, Alejandro Abdelnur and Elaine Chien, Sun Microsystems
- Multiple sub-pages (portlets) aggregated into a single displayed page (portal page)

Agenda

- Introduction
- JavaServer™ Faces Technology and Portlets
  - What do we have?
  - What do we need?
  - Customization and implementation
- Creating your JSF Portlet
- Demo

JavaServer™ Faces Technology and Portlets

What Support for Portlets Is in JSF Today?

- Full support for servlets
  - Container is assumed to be servlet
  - Core API and RI enabled for portlets
- Container-specific code isolated in ExternalContext
- No reference implementation of portlet support

JavaServer™ Faces Technology and Portlets

What Support for Portlets Is Required?

- Providers of Portal Servers each provide their own bespoke solution
- Write your own—requires:
  - New Portlet-based ExternalContext class
  - Factory to create this ExternalContext
  - Portlet container, providing support equivalent to that available for Servlet

Implementation

Very Few Classes Needed

- JavaServer™ Faces technology is “portlet-ready”
- Container-specific code isolated in ExternalContext
  - Portlet implementation of ExternalContext
  - Factory to create the ExternalContext
- Wrapper for portlet container
  - Corresponds to servlet
  - Provide default processing
- We can reuse the standard JSF lifecycle

Implementation

FacesContextFactory

- getFacesContext returns a FacesContext
- Simple implementation:
  - Creates a new Portlet-based ExternalContext
  - Creates a new FacesContext based on this

Implementation

FacesContextFactory

```
public class PortletFacesContextFactory {
   // Simplified creation of Portlet’s FacesContext
   public FacesContext getFacesContext(
                        PortletContext context,
                        PortletRequest request,
                        RenderResponse response,
                        Lifecycle lifecycle)
                        throws FacesException {

      FacesContext fc = new FacesContextImpl(
                  new PortletExternalContext(
                        context,
                        request,
                        response),
            lifecycle);
      return fc;
   }
}
```

Implementation

FacesContextFactory

- More complete implementation
  - Because of portlet lifecycle, we need to store intermediate context
  - Check if context already in the request
  - Store the context in the request

Implementation

FacesContextFactory

```
FacesContext fc =
   (FacesContext) request.getAttribute(
                              FACES_CONTEXT_ATTR);
if (null != context) {
   fc = new FacesContextImpl(
               context.getExternalContext(),
               lifecycle);
} else {
   fc = new FacesContextImpl(
         new PortletExternalContextImpl(
               context, request, response),
               lifecycle);
}
request.setAttribute(FACES_CONTEXT_ATTR, fc);
```

Implementation

PortletExternalContext

- Isolates container-dependant code
- Defined in section 6.1.2 of JSF spec
  - Too much detail to cover here
- Cannot extend servlet-based ExternalContext
  - Based on portlet, not servlet
  - Watch out for portlet/servlet differences
    - e.g. Dispatch must use *include* and never *forward*

Implementation

PortletExternalContext

```
// Portlet cannot use forward – use include instead
public void dispatch(String requestURI)
      throws IOException, FacesException {

   PortletRequestDispatcher requestDispatcher =
      portletContext.getRequestDispatcher(
                                 requestURI);
   requestDispatcher.include((RenderRequest)
                  this.request, this.response);
}
```

Implementation

Portlet Container

- Replaces Servlet as container
- Extends javax.portlet.GenericPortlet
- Encapsulates default behaviours
- Maps portlet lifecycle onto JSF lifecycle

Implementation

Portlet Container

- Special considerations for portlet
  - JSF Lifecycle split into two parts
    - Execute: request-processing phases
    - Render: GUI generation
  - Portlet has two lifecycles
    - Action
      - No change to display
      - Calls JSF request-processing phases only
    - Render
      - Updates display
      - Calls JSF request-processing phases
      - Calls JSF render phase too

Implementation

Portlet Container—Actions

```
public void processAction(ActionRequest request,
                          ActionResponse aResponse)
      throws PortletException {

   // Acquire Lifecycle instance
   Lifecycle lifecycle = getLifecycle(request);

   // Acquire the FacesContext
   FacesContext context = getFacesContext(
      (PortletRequest) request, null, lifecycle);

   // Pick up the current mode
   PortletMode mode = request.getPortletMode();

   executeAction(request, lifecycle,
                 context, mode.toString);
}
```

Implementation

Portlet Container—Actions

```
private void executeAction(ActionRequest request,
                           Lifecycle lifecycle,
                           FacesContext context,
                           String mode)
                        throws PortletException {
   // Restore info about current state
   restorePage(context, request, mode);

   // Execute the JSF request-processing lifecycle
   try {
      lifecycle.execute(context);
   } catch (FacesException e) {
      throw new PortletException(e.getMessage(),e);
   }

   // Save info about current state
   savePage(context, mode);
}
```

Implementation

Portlet Container—Render

```
protected void doRender(RenderRequest request,
                        RenderResponse response,
                        String mode)
      throws IOException, PortletException {
   // Acquire Application & Lifecycle
   Application application = getApplication();
   Lifecycle lifecycle = getLifecycle(request);
   // Restore info about current state
   restorePage(context, request, mode);

   // Call JSF lifecycle: process AND render
   lifecycle.execute(context);
   lifecycle.render(context);

   // Save current info & release context
   savePage(context, mode);
   context.release();
}
```

Implementation

Portlet Container—Additional Processing Needs

- Render variants
  - Modes defined in deployment descriptor
    - doEdit
    - doView
    - doHelp
  - Also possible custom modes (doConfig, etc.)
- Current page/view and its mode associations
  - Save
  - Restore

Implementation

Portlet Container—Render Variants/Modes

```
// View mode processing
public void doView(RenderRequest req,
                   RenderResponse resp)
            throws PortletException, IOException {
   doRender(req, resp, VIEW_MODE);
}
// Edit mode processing
public void doEdit(RenderRequest req,
                   RenderResponse resp)
            throws PortletException, IOException {
   doRender(req, resp, EDIT_MODE);
}
// Help mode processing
public void doHelp(RenderRequest req,
                   RenderResponse resp)
            throws PortletException, IOException {
   doRender(req, resp, HELP_MODE);
}
```

Implementation

Portlet Container—Save and Restore

```
private void savePage(FacesContext context,
                      String mode) {
   String page = context.getViewRoot().getViewId();
   Map sessionMap =
      context.getExternalContext().getSessionMap();
   sessionMap.put(PAGE_ATTR + mode, page);
}
```

Implementation

Portlet Container—Save and Restore

```
private void restorePage(FacesContext context,
                         PortletRequest request,
                         String mode) {
   Object page = null;
   // set up previous or initial page for this mode
   Map sessionMap = context.getExternalContext()
                           .getPortletSessionMap();
   if (sessionMap != null) {
      page = sessionMap.get(PAGE_ATTR + mode);
      // Try hardcoded value
      if (page == null) {
         page = getPortletConfig()
            .getInitParameter(PAGE_ATTR + mode);
      }
      sessionMap.put(PAGE_ATTR + mode, page);
   }
}
```

Agenda

- Introduction
- JavaServer™ Faces Technology and Portlets
- Creating your JSF Portlet
  - Application management
  - Page contents
- Demo

Creating a JSF Portlet

Application Management

- Faces configuration file (faces-config.xml)
  - Point at the correct Faces Context factory
- Portlet deployment descriptor (portlet.xml)
  - Specify portlet class
  - Define modes and their associated initial pages

Creating a JSF Portlet

Faces Configuration File: faces-config.xml

```
<faces-config>
   <factory>
      <faces-context-factory>
            my.faces.PortletFacesContextFactory
      <faces-context-factory>
   </factory>
</faces-config>
```

Creating a JSF Portlet

Portlet Deployment Descriptor: portlet.xml

```
<portlet>
   <portlet-name>MyFacesPortlet</portlet-name>
   <display-name>JavaOne JSF Portlet</display-name>
   <portlet-class>
      my.faces.FacesGenericPortlet
   </portlet-class>
   <init-param>
      <name>my.page.view</name>
      <value>/MyFacesView.jsp</value>
   </init-param>
   <init-param>
      <name>my.page.edit</name>
      <value>/MyFacesEdit.jsp</value>
   </init-param>
   ...
</portlet>
```

Creating a JSF Portlet
Page Contents

- Include tag libraries
  - JSF core tags
  - JSF HTML tags
  - Any other special tags
- The page has no <head>, <title>, <body> tags
- Encapsulate the markup in <view> … </view>
- Any references must be encoded
  - Call renderResponse.encodeURL() on path
  - Path made from RenderRequest.getContextPath() prepended to file path
- Create the page as a normal JSF page

Creating a JSF Portlet

Simple Page—Contains a Single Submit Button

```
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
<%@taglib uri="http://java.sun.com/portlet" prefix="portlet"%>
<portlet:defineObjects />
<f:view>
   <link rel="stylesheet" type="text/css"
      href='<%= renderResponse.encodeURL(
                  renderRequest.getContextPath() +
                  "/theme/stylesheet.css") %>'
      title="Style">
   <h:form styleClass="form" id="form1">
      <h:commandButton type="submit" value="Submit"
                  id="button1"></h:commandButton>
   </h:form>
</f:view>
```

Demo

Creating and Running a Portlet Using JavaServer™ Faces Technology

Summary

- JSF in Portlets needs infrastructural support
  - ExternalContext
  - Portlet container
- Portlet application requires customization of
  - Faces configuration file
  - Portlet deployment descriptor
- The rest is easy …J

For More Information

- JSF Links
  - Sun—http://java.sun.com/j2ee/javaserverfaces/
  - JSF Central—http://www.jsfcentral.com
- JSF Books
  - JavaServer Faces In Action—Kito Mann (Manning)
  - JavaServer Faces—Hans Bergsten (O’Reilly)
  - Core JavaServer Faces—David Geary (Sun)
- JSF and Portlets
  - I couldn’t find any doc. Anywhere!

Q&A

Brendan Murray, IBM

JavaOne℠

Sun's 2004 Worldwide Java Developer Conference℠

Putting Faces on Your Portlets

Exploiting JavaServer™ Faces Technology in Portlet Applications

Brendan Murray

Senior Software Architect
IBM Dublin Software Lab

http://www.ibm.com