IBM Software Group

Introduction to Portlet Development

WebSphere Portal Server v5.1

Lotus software

Product Introduction + Exploration

ON DEMAND BUSINESS

IBM Proof of Technology

IBM Product Introduction + Exploration

Session Objectives

- After completing this session, you should be able to:
  - Understand the differences between IBM Portlet API and JSR 168
  - Understand the capabilities of both IBM Portlet API and JSR 168
  - Understand Portlet Development

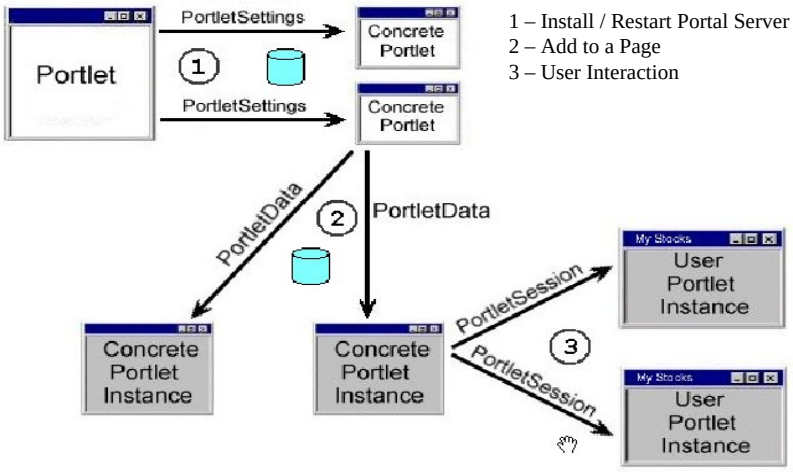
Agenda

- Definition of a Portlet
  - Portlet Manifestations
- Portlet API
  - IBM Portlet API
  - JSR168
  - IBM Portlet API and JSR 168 Comparison
- Portlet Applications
  - IBM Portlet API
  - JSR 168
- Portlet Development Environment
  - What is needed ?
  - Rational Application Developer 6.0
- Summary and Questions

What is a portlet?

- A Portlet is a "plugable" component that represents an application
- From a developer's perspective, it is a Java client that runs on the server
- Provides output to the user by generating markup output that is assembled into a portal page by the portal container
- Manages the user's preferences for the associated application

Manifestations of a Portlet



1 – Install / Restart Portal Server
2 – Add to a Page
3 – User Interaction

Note: PortletSettings and PortletData are PortletPreferences in JSR168 API

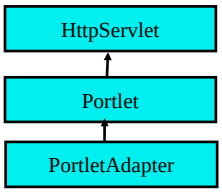
Portal Configuration Objects

Object Name	Portlet Type	R/O vs. R/W	Element name and or API call	Purpose
PortletConfig	Abstract Portlet	R/O	<**init-param**> in web.xml ▪Argument for *init*() ▪*getServletConfig*()	Stores initial static config from Portlet Developer
PortletSettings	Concrete Portlet	R/O, R/W in Cfg mode	<**config-param**> in portlet.xml *PortletRequest.getPortletSettings*()	Stores concrete portlet dynamic data
PortletApplicationSettings	Concrete Portlet Application	R/O, R/W in Cfg Mode	<**context-param**> in portlet.xml *PortletSettings.getApplicationSettings*	Stores common information about the portlet app
PortletData	Concrete Portlet Instance	R/O, R/W in Edit Mode	*PortletRequest.getData*()	Stores persistent concrete portlet data
PortletContext	Abstract Portlet Application	R/O	<**context-param**> in web.xml	Portlets' view of the container
PortletSession	User Portlet Instance	R/W	*PortletRequest .getSessionData*()	Stores user specific transient data

IBM Portlet API

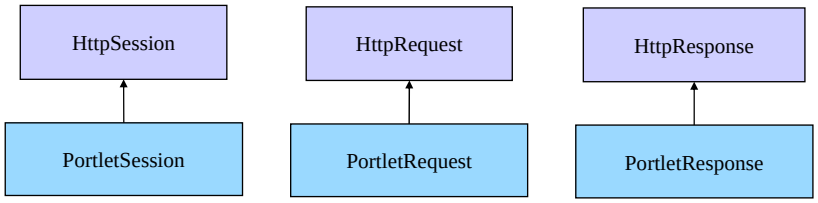
Portlet API Inheritance

- Portlet API inherits directly from HttpServlet.
- It is not an independent entity.

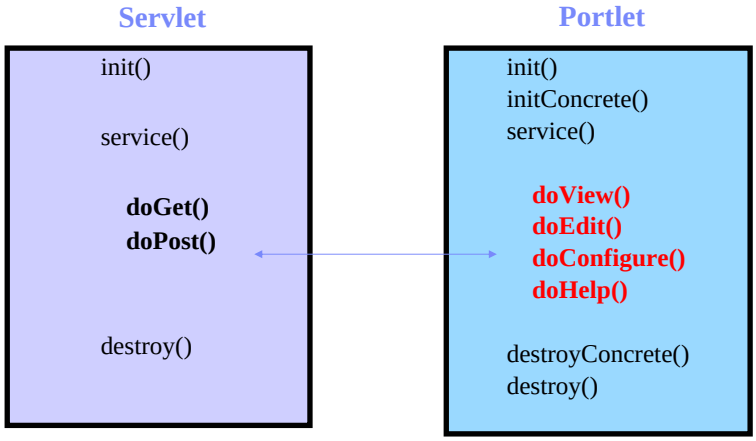


Portlet Configuration Inheritance

- Portlet configuration objects now inherit from their corresponding HTTP counterpart.



Portlet API Comparison



Comparing Portlets with Servlets

- Portlets (as servlets) are web components
- Portlet API is modeled after Servlet API
- Portlet lifecycle is similar to servlet lifecycle (init, request processing, destroy)
- Portlets decouple processing from rendering
- Portlets have additional state: portlet mode, window state, render parameters, portlet preferences

Portlet Applications

- Portlet applications are Web applications
- All HttpSession listeners can be used
  - Utility class to parse namespaced session attributes (Portlet entity scope)

```
PortletSessionUtil.decodeAttributeName(String);
```

- Two deployment descriptors
  - web.xml
    - Web app settings
    - Servlets
    - JSPs
    - Security roles
    - …
  - portlet.xml
    - Portlet definitions
    - Formally defined through XML schema

JSR 168 API

Prior to JSR 168

- Different portal vendors have defined different APIs
  - Different APIs for local “components”
    (simple JSPs / portlets / channels / modules / ...)
- No interoperability between local portlets and portal servers
- Application and content providers must implement different portlets for different portal servers
- Customers developing significant numbers of portlets are quickly locked into a particular portal solution

Current Standards

- Portlet standard for local Java portlets
  - Java Portlet Specification, first version defined as JSR 168
  - Co-led by IBM and Sun
  - Reference implementation available at Apache, provided by IBM
    - http//jakarta.apache.org/pluto
  - Compliance test suite available from Sun
- Portlet standard for remote portlets
  - Web Service for Remote Portlets (WSRP), defined at OASIS
  - Chaired by IBM
  - Sample implementation available at Apache, provided by IBM
    - http//ws.apache.org/wsrp4j
  - Conformance test suite available at alphaworks
    - http://alphaworks.ibm.com/tech/wsrptk

JSR 168 Goals and Scope

- JSR 168 goals
  - Simple programming model
  - Portability
  - Leverage and align with J2EE technologies
  - Interoperability with other Java technologies
- JSR scope
  - Portlet API
  - Portlet container
  - Contract between the API and the container
  - Deployment unit, portlet application

JSR 168 Portlet Container

- **Extension** of the servlet container
- It handles portlet components in addition to servlet and JSP components
- It shares much of the servlet container functionality (HTTP request handling, web application context, class loaders, session management, security)

JSR 168 Portlet features:

- Portlet Interface

```
public interface Portlet {
  public void init(PortletConfig) throws ...;
  public void processAction
  (ActionRequest,ActionResponse) throws ...;
  public void render
  (RenderRequest,RenderResponse) throws ...;
  public void destroy();
}
```

- Portlet Request Types
  - Action request

    processAction (ActionRequest, ActionResponse)
    - Invoked only when the URL is an action targeted to the portlet, it does not produce output
  - Render request

    render (RenderRequest, RenderResponse)
    - Invoked on every request to the portal, it produces output to create the portal page
    - Portal keeps resending last set of render parameters

JSR 168 Portlet features:

- Portlet URLs
  - Portlets are always accessed through a portal. Portlets do not have a URL mapping
  - Portlet URLs allow portlets to create URLs that target to themselves (through the portal end-point)
    - ActionURL – processAction method of portlet is called
      - RenderResponse.createActionURL()
    - RenderURL – new render parameters are set, only render is called
      - RenderResponse.createRenderURL()
- Portlet Preferences
  - Persistent read/write portlet configuration managed by the portlet container
    - Write is only allowed in action
  - Normally, portlet preferences are per portlet/per user
  - Two types of preferences
    - Read-only – can only be changed by the administrator (e.g. in Config mode)
      - Example: Stock quote server of a stock quote portlet
    - Writable – can be changed by the user (e.g. in the Edit mode)
      - Example:company stock symbols of a stock quote portlet
  - Default values are defined in the portlet.xml deployment descriptor

Caching

- Expiration based caching
- Default expiration time can be set in the deployment descriptor
- Portlet can change the expiration time programmatically
- Cache is invalidated at expiration time or when the portlet is the target of an action

IBM Portlet API and JSR 168 Comparison

Supporting two Portlet APIs

- WebSphere Portal 5.1 supports two portlet containers
  - JSR 168 API
  - IBM Portlet API
- Portlets of both containers can be on the same page
- Transparent for the users of the portal
- Same deployment and administration for both portlet types

Portlet Programming Tools Support for WP 5.1

Portlet API	Framework	SDO (*1)	Cooperative portlet using wiring	Cooperative portlet using Click-to-Action	Collaborative portlet (People Awareness)
IBM WP API	None	Yes (*2)	Yes	Yes	Yes
	Struts	Yes (*2)	Yes	Yes	Yes
	JSF	Yes	Yes	Yes	Yes
JSR 168	None	Yes (*2)	Yes	No	Yes
	Struts	Yes (*2)	Yes	No	Yes
	JSF	Yes	Yes	No	Yes

*1) SAP/Siebel only. JBDC is provided for prototype only, not for production use. No IBM service is provided.

*2) Some coding is required.

No tool support in 6.0 (post 6.0)
WP 5.1 supports Struts JSR 168.

IBM Portlet API compared to JSR 168

- Basic concepts are the same
- JSR 168
  - General API of choice for creating new applications
  - Reconsider use when application requires
    - Legacy inter-portlet messaging
    - C2A
    - Struts tooling
- IBM Portlet API
  - Continued support well into the future

Equal Concepts

- Portlet Modes
  - View, Help, Edit, (Configure)
- Window States
  - Normal, Minimized, Maximized
- Portlet life cycle and request processing
  - Init, handle requests, destroy
- URL encoding
- Include servlets and JSPs
  - Only minor difference: multiple markup and device search of WebSphere Portal is not available in the JSR 168
- Portlet application packaging
- Expiration-based caching
  - Difference: JSR 168 cannot set same cache entry for all users

Different Concepts: PortletEntity

- JSR 168
  - PortletEntity that is parameterized by PortletPreferences
    - The same administrative user interface
    - Portlet only has one object to set perferences.
      - Config-Mode sets administrative preferences (→ concrete portlet)
      - Edit-Mode sets user preferences (→ concrete portlet instance)
- WebSphere Portal
  - Portlet parameterized by PortletSettings (concrete portlet)
    - done by the admin
  - Concrete portlet personalized by PortletData (concrete portlet instance)
    - done by users

Different Concepts: Request and response

- JSR 168
  - Request and response objects are different for action and render phase.
  - Cleaner programming model as render should be re-playable.
  - Aligned with WSRP.
- WebSphere Portal
  - Request and response objects are the same for action and render phase.
  - Portlets may store attributes in the action phase in the request and retrieve it in the render phase. In JSR 168, the portlet must explicitly make call(s) to setRenderParameter() to provide attributes to render phase.

New JSR 168 Concepts (1 of 3)

- Render parameters and navigational state
  - Stay the same between render requests
  - Allow the portlet to store its navigational state
  - Enables bookmarkability and allows to use the browser back button
- Extension mechanisms
  - Custom portlet modes, custom window states
  - Custom user information
  - Request and response properties
- Reuse of the HttpSession listeners
  - Utility class allows decoding of namespacing

New JSR 168 Concepts (2 of 3)

- J2EE role support
- Localization support
  - Resource bundles
  - on deployment descriptor (DD) level
    - xml:lang attributes for all descriptions, display names
  - on portlet level
    - access to resource bundle defined in DD or portlet-info
- Multiple response content types
  - allows transcoding
- Redirect in performAction

New JSR 168 Concepts (3 of 3)

- Preference validator
  - Allows to separate the validation logic from the portlet
  - May also be used by external tools
- Portal context
  - Provides information about the portal
    - Supported portlet modes / window states
    - Properties (extensions)
- Aligned with WebServices for Remote Portlets (WSRP)
  - Standard at OASIS (http://www.oasis.org)
  - JSR 168 portlets can be exported as WSRP services
  - A JSR 168 proxy portlet can proxy WSRP services

Portlet API Objects

JSR 168 API	IBM Portlet API
ActionRequest	PortletRequest
RenderRequest	
ActionResponse	PortletResponse
RenderResponse	
PortletSession	PortletSession
PortletConfig	PortletConfig
PortletContext	PortletContext
PortletPreferences	PortletData
	PortletSettings
PortalContext	
Request.getAttributeMap(xyz)	User

Tag Library

JSR 168 API

- defineObjects
- actionURL / renderURL
  - param
- JSTL recommended for common tasks (resource bundle access)

IBM Portlet API

- init
- create(Return)URI
  - URIParameter
  - URIAction
- encodeNamespace
- client
- if
- dataAttribute/Loop
- settingsAttribute/Loop
- log
- JSTL recommended for common tasks (resource bundle access)

Portlet Applications

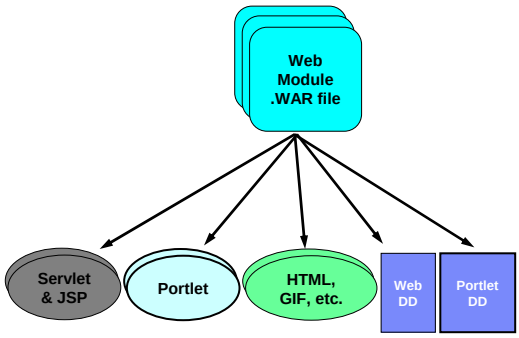
Packaging Portlets

- Create the directory structure for your portlet application.
- The portlet application directory structure is very similar to the directory structure for a web application
- This directory structure is maintained in the portlet application WAR file:

/WEB-INF/web.xml
/WEB-INF/portlet.xml
/WEB-INF/lib/portletApp.jar
/WEB-INF/classes/portletClass2.class
/jsp/portletViewJsp.jsp
/images/portletLogo.gif

Packaging Portlet Applications

- Portlets packaged in Web Modules (WAR files)
- Portlet WAR file is deployed on the Portal Server



Deployment Descriptors

A portlet application is defined to the portal with two XML documents included in the /WEB-INF directory of the WAR file

- web.xml
  - Defines the web application characteristics of the portlet application. This includes portlet class names and portlet configuration data
- portlet.xml
  - Defines the portal server characteristics of the portlet application. This includes portlet application names, portlet titles, and other portlet configuration data

Packaging

JSR 168 API

- Two deployment descriptors
  - web.xml
  - portlet.xml
- They do not depend on each other
  - All portlet related deployment information is stored in the portlet.xml only
- Abstract and Concrete collapsed (PortletEntity)

IBM Portlet API

- Two deployment descriptors
  - web.xml
  - portlet.xml
- They depend on each other
  - Each portlet must have a corresponding servlet defined in the web.xml
- Portlet Deployment Descriptor defines abstract and concrete portlets

Linking Servlet, Portlet, and Concrete Portlet (IBM Portlet API):

web.xml

```
<web-app id="WebApp_1">
  <display-name>Mail</display-name>
  <Servlet id="Servlet_1">
    <Servlet-name>MailPortlet</Servlet-name>
    <Servlet-class>MyMailPortlet</Servlet-class>
    <init-param id="InitParam_1">
      <param-name>controller.html</param-name>
      <param-value>HtmlController</param-value>
    </init-param>
  </Servlet>
  <Servlet id="Servlet_2">
    <Servlet-name>CalendarPortlet</Servlet-name>
    <Servlet-class>MyCalPortlet</Servlet-class>
    <init-param id="InitParam_2">
      <param-name>controller.html</param-name>
      <param-value>HtmlController</param-value>
    </init-param>
  </Servlet>
  <Servlet-mapping id="ServletMapping_1">
    <Servlet-name>MailPortlet</Servlet-name>
    <url-pattern>/MailPortlet/*</url-pattern>
  </Servlet-mapping>
  <Servlet-mapping id="ServletMapping_2">
    <Servlet-name>CalendarPortlet</Servlet-name>
    <url-pattern>/CalendarPortlet/*</url-pattern>
  </Servlet-mapping>
</web-app>
```

Portlet.xml

```
<Portlet-app-def>
  <Portlet-app
uid="com.ibm.wps.samplet.mail.4969">
    <Portlet-app-name>NotesSuite</Portlet-app-
name>
    <Portlet id="Portlet_1"
          href="WEB-INF/web.xml#Servlet_1">
        <Portlet-name>Mail</Portlet-name>
        ... </Portlet>
    <Portlet id="Portlet_2"
          href="WEB-INF/web.xml#Servlet_2">
        <Portlet-name>Calendar</Portlet-name>
        ... </Portlet>
  </Portlet-app>
  <concrete-Portlet-app
          uid="com.ibm.wps.samplet.mail.4969.1>
    ...
    <concrete-Portlet href="#Portlet_1">
        ... </concrete-Portlet>
    <concrete-Portlet href="#Portlet_2">
        ... </concrete-Portlet>
  </concrete-Portlet-app>
  <concrete-Portlet-app>
    ...
    <concrete-Portlet href="#Portlet_1">
        ... </concrete-Portlet>
  </concrete-Portlet-app>
</Portlet-app-def>
```

Portlet Development Environment

Portlet Development Environment

- Install the required portlet development tools
  - Java JDK 1.3 (installed with WAS 5.1)
  - Java Source Editor
  - HTML/JSP Editor
- Add the required and optional JAR files to your Java CLASSPATH
- <WAS-HOME>/lib/app
  - portlet-api.jar – Portlet API interfaces and classes
  - wps.jar – Portal base interfaces and classes
  - wpsportlets.jar – Generic & Helper Portlets
- <WAS-HOME>/lib
  - j2ee.jar – Servlet API and J2EE interfaces and classes
  - websphere.jar – for Cacheable portlet

Session Summary

- In this unit you have been introduced to:
  - IBM Portlet API
  - JSR 168 API standards
  - The Support for two Portlet Containers and the differences between the their APIs
  - The new API extension features for JSR 168 based portlets

Lab 2.2 Exercise: Portlet Development Lab Overview

- Exercise one
  - Develop a Portlet using the IBM Portlet API
    - Portlet will access DB2 database
    - Portlet will incorporate People Awareness
- Exercise Two
  - Develop a Portlet using the JSR 168 Specification
  - Portlet will access a DB2 database