Lotus Expeditor 6.1 Education

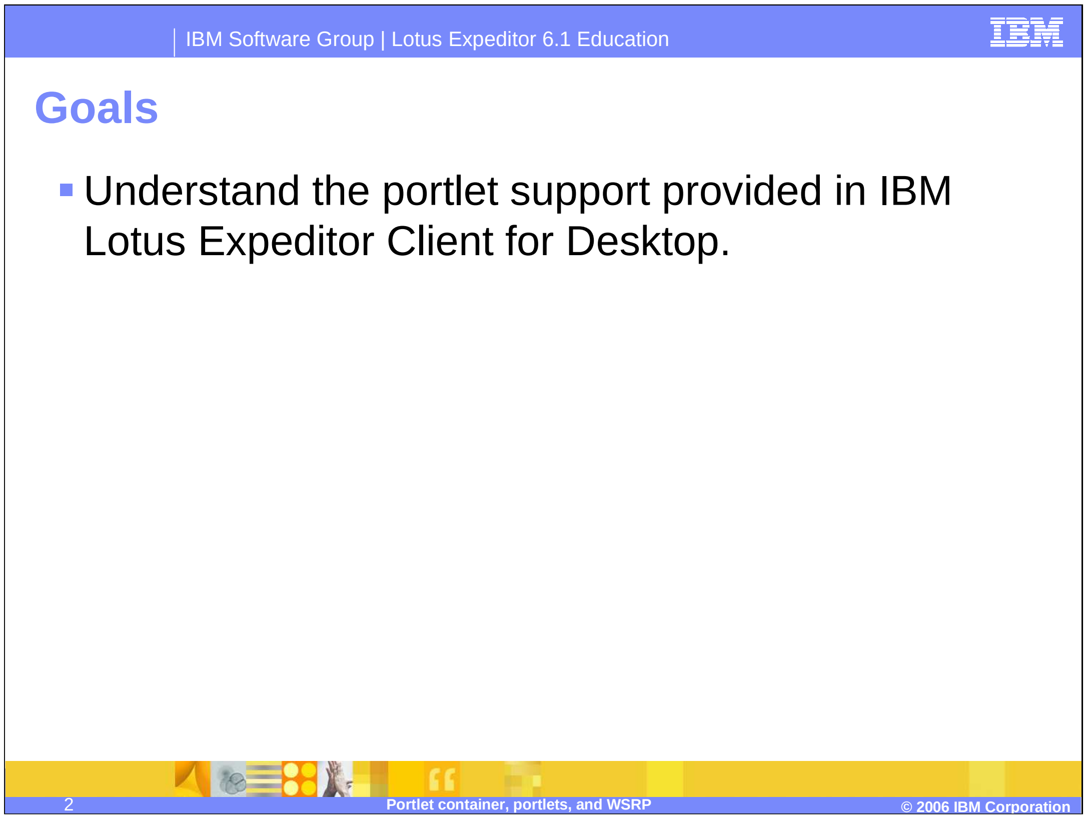
IBM® Lotus® Expeditor 6.1 Client for Desktop

Portlet Container, Portlets and WSRP

Lotus software

business on demand software

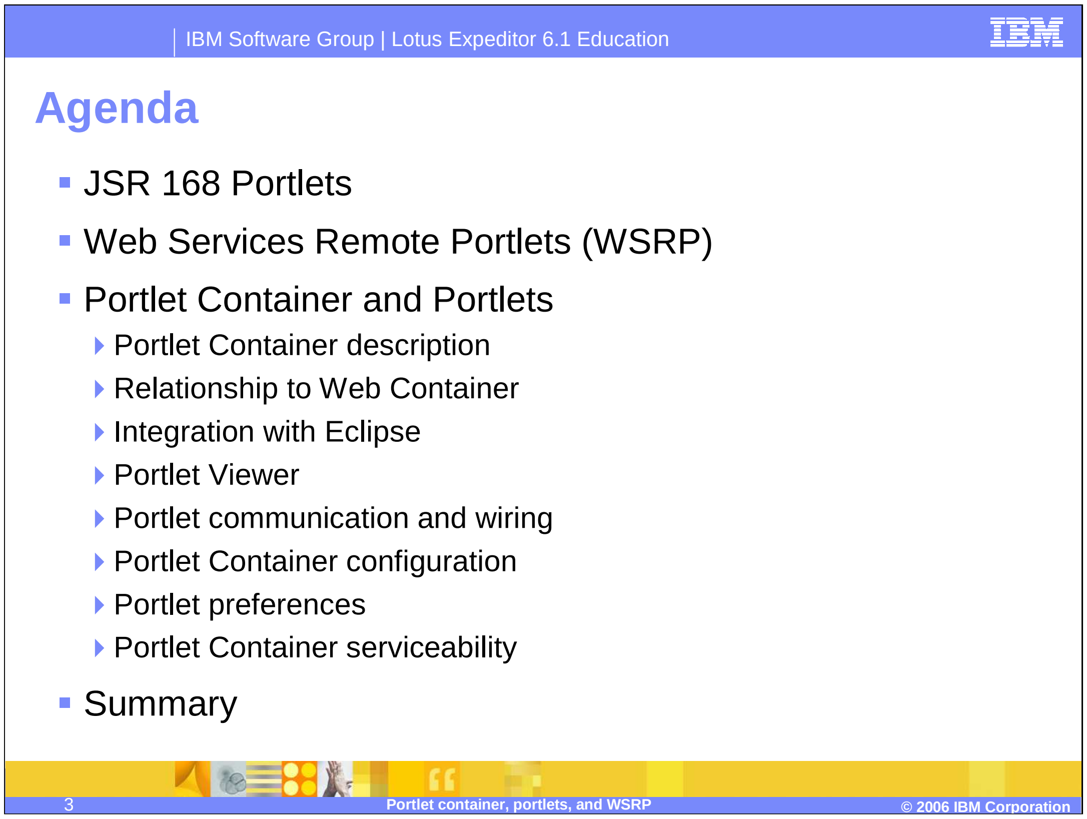
This presentation explains Portlet support in IBM Lotus Expeditor 6.1 Client for Desktop.



Goals

- Understand the portlet support provided in IBM Lotus Expeditor Client for Desktop.

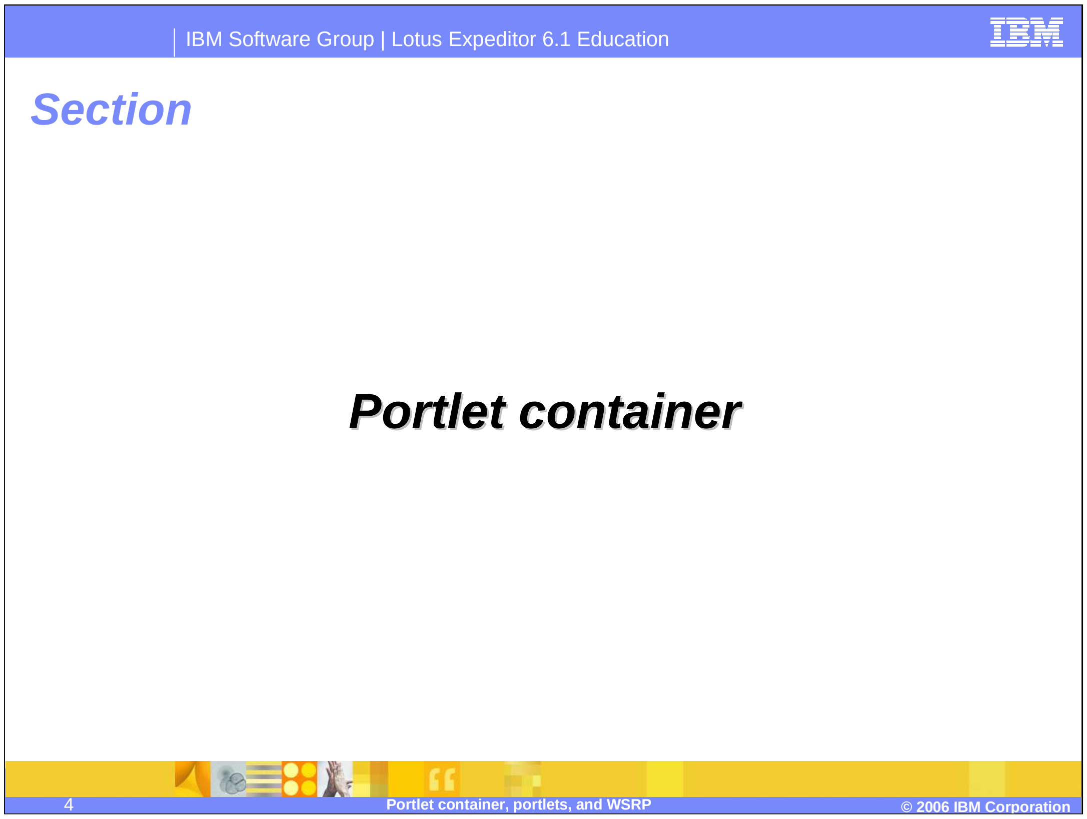
The goal of this presentation is to understand the portlet support provided in IBM Lotus Expeditor 6.1 Client for Desktop.



Agenda

- JSR 168 Portlets
- Web Services Remote Portlets (WSRP)
- Portlet Container and Portlets
  - Portlet Container description
  - Relationship to Web Container
  - Integration with Eclipse
  - Portlet Viewer
  - Portlet communication and wiring
  - Portlet Container configuration
  - Portlet preferences
  - Portlet Container serviceability
- Summary

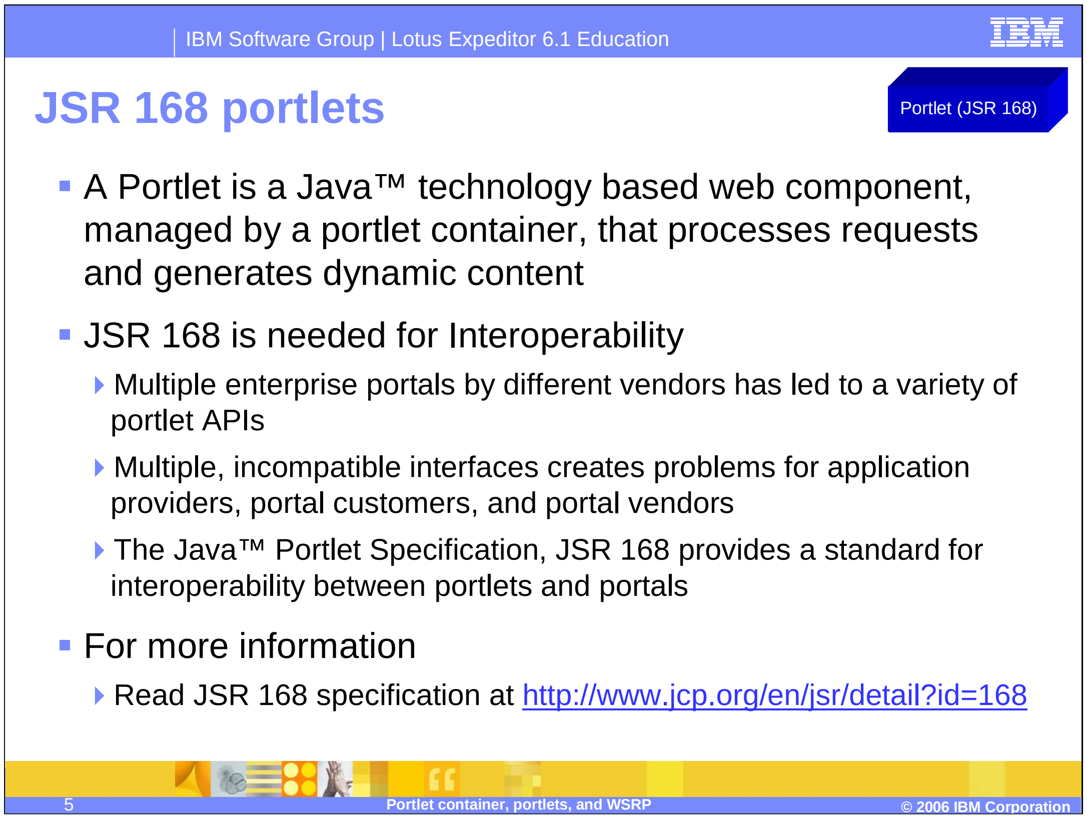
The agenda of this presentation is to explain the Portlet Application capabilities the client platform provides to you, the infrastructure and plug-ins that enable these capabilities, and details about the Portlet Applications and Portlet Container supported by IBM Lotus Expeditor 6.1 Client for Desktop.



Section

Portlet container

Let’s describe more details about the Portlet Container.



IBM Software Group | Lotus Expeditor 6.1 Education

JSR 168 portlets

Portlet (JSR 168)

- A Portlet is a Java™ technology based web component, managed by a portlet container, that processes requests and generates dynamic content
- JSR 168 is needed for Interoperability
  - Multiple enterprise portals by different vendors has led to a variety of portlet APIs
  - Multiple, incompatible interfaces creates problems for application providers, portal customers, and portal vendors
  - The Java™ Portlet Specification, JSR 168 provides a standard for interoperability between portlets and portals
- For more information
  - Read JSR 168 specification at http://www.jcp.org/en/jsr/detail?id=168

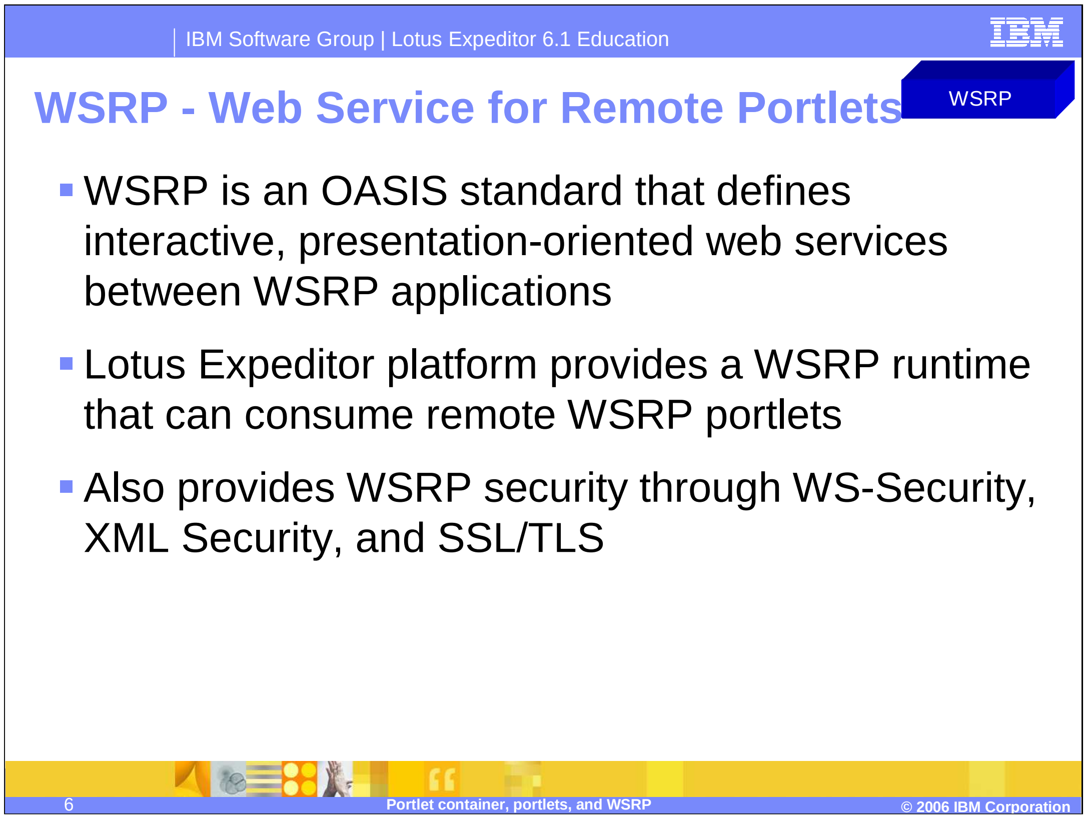
5 Portlet container, portlets, and WSRP © 2006 IBM Corporation

First let's discuss Portlets.

A Portlet is a Java technology based Web component managed by a portlet container that processes requests and generates dynamic content.

The JSR 168 portlet specification defines a standard API for portlet development, and it enables interoperability among portlets and portals.

Without this standard, each version of an application needed its own Portlet API, and each of the various portals required that these portlets be specifically tailored for implementation through that portal.



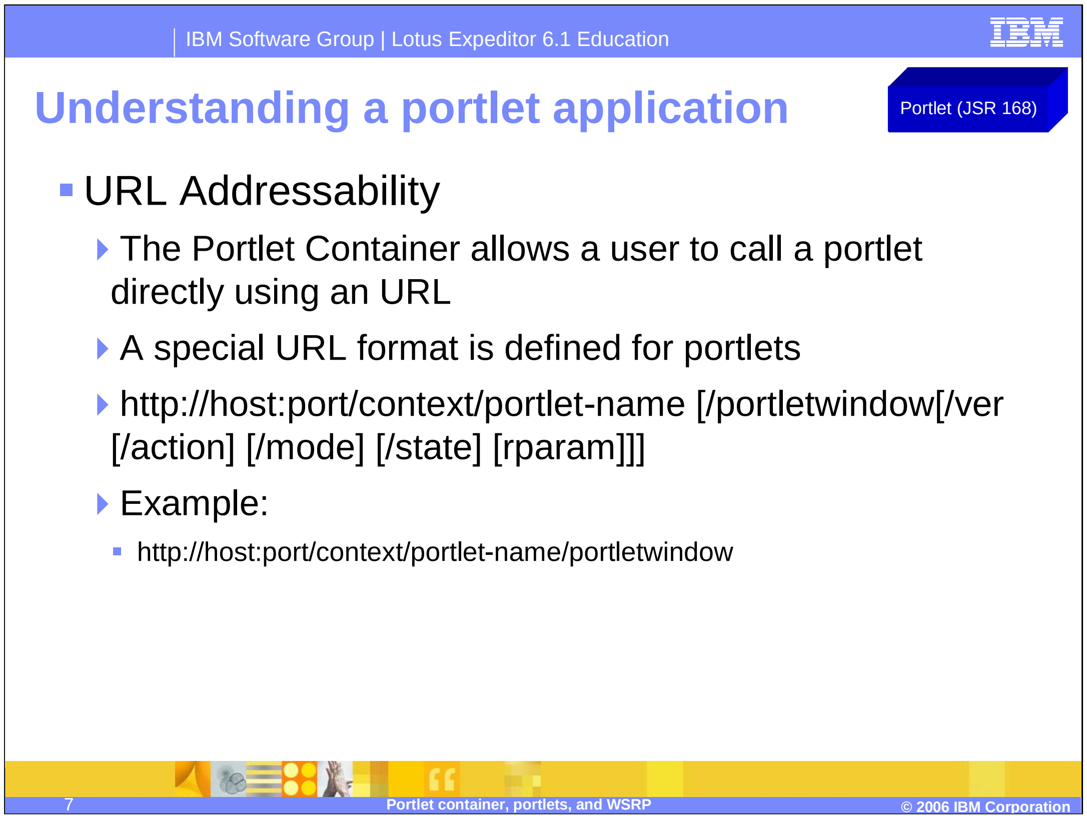
WSRP - Web Service for Remote Portlets

WSRP

- WSRP is an OASIS standard that defines interactive, presentation-oriented web services between WSRP applications
- Lotus Expeditor platform provides a WSRP runtime that can consume remote WSRP portlets
- Also provides WSRP security through WS-Security, XML Security, and SSL/TLS

WSRP is an OASIS standard that defines interactive, presentation-oriented Web services between WSRP-enabled applications, such as the Portal server or the WSRP runtime within the Lotus Expeditor. The WSRP runtime retrieves portlet content from remote portlets and processes user interaction.

Through the WSRP runtime, the portlet viewer is used to support presentation and interaction with remote portlets inside a portal server.


IBM Software Group | Lotus Expeditor 6.1 Education

Understanding a portlet application

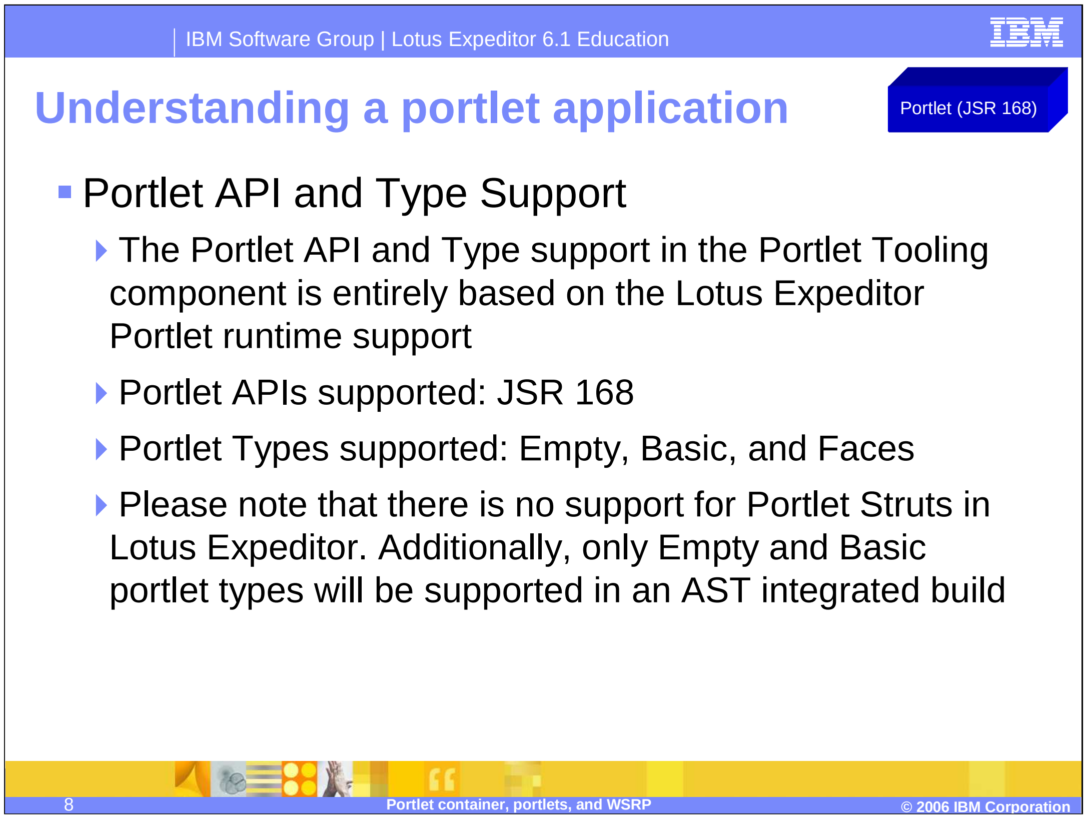
Portlet (JSR 168)

- URL Addressability
  - The Portlet Container allows a user to call a portlet directly using an URL
  - A special URL format is defined for portlets
  - http://host:port/context/portlet-name [/portletwindow[/ver [/action] [/mode] [/state] [rparam]]]
  - Example:
    - http://host:port/context/portlet-name/portletwindow

7 Portlet container, portlets, and WSRP © 2006 IBM Corporation

Next let’s discuss URL Addressability for portlets.

The Portlet Container allows a user to call a portlet directly using a URL. Therefore, a special URL format is defined for portlets. This URL format allows portlets to be called just like servlets (by context and portlet-name), but also provides the possibility to add additional portal context information, such as the portlet mode or window state.



Understanding a portlet application

Portlet (JSR 168)

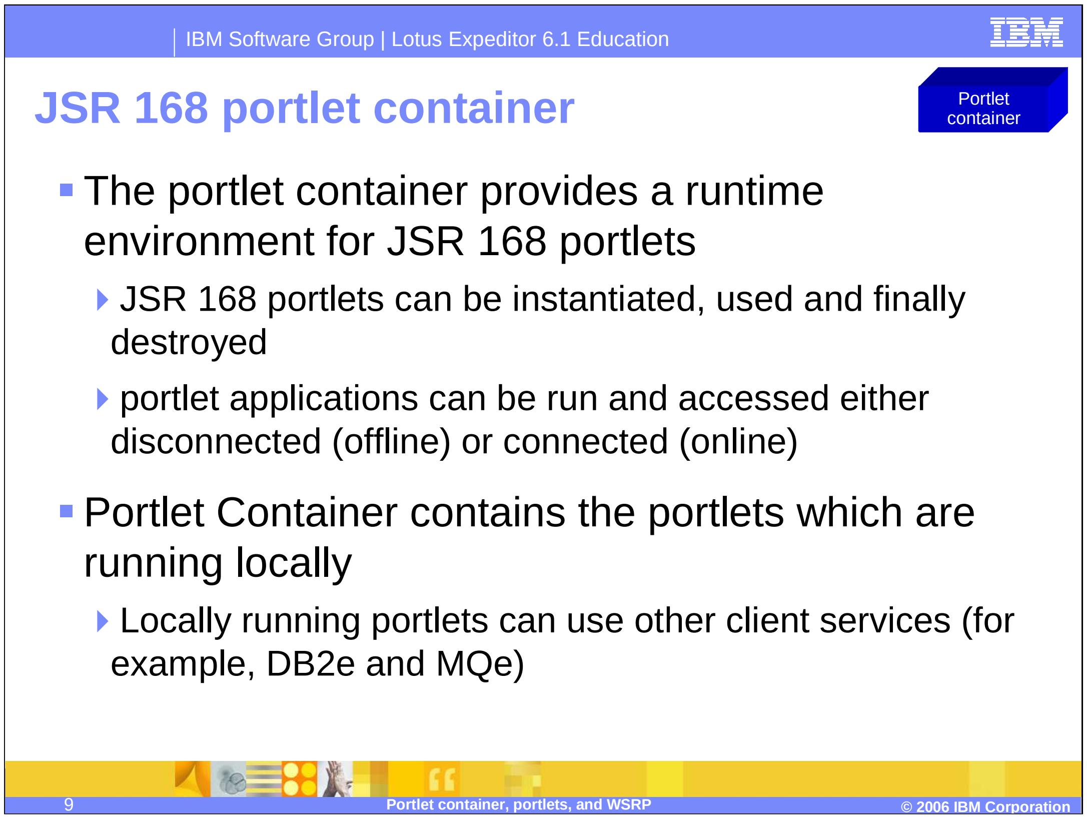
- Portlet API and Type Support
  - The Portlet API and Type support in the Portlet Tooling component is entirely based on the Lotus Expeditor Portlet runtime support
  - Portlet APIs supported: JSR 168
  - Portlet Types supported: Empty, Basic, and Faces
  - Please note that there is no support for Portlet Struts in Lotus Expeditor. Additionally, only Empty and Basic portlet types will be supported in an AST integrated build

The Portlet API and Type support in the Portlet Tooling component is entirely based on the Lotus® Expeditor Portlet runtime support.

JSR 168 Portlet APIs are supported.

The Portlet types supported are Empty, Basic, and Faces.

Note that there is no support for Portlet Struts in Lotus Expeditor. Additionally, only Empty and Basic portlet types will be supported in an AST integrated build.



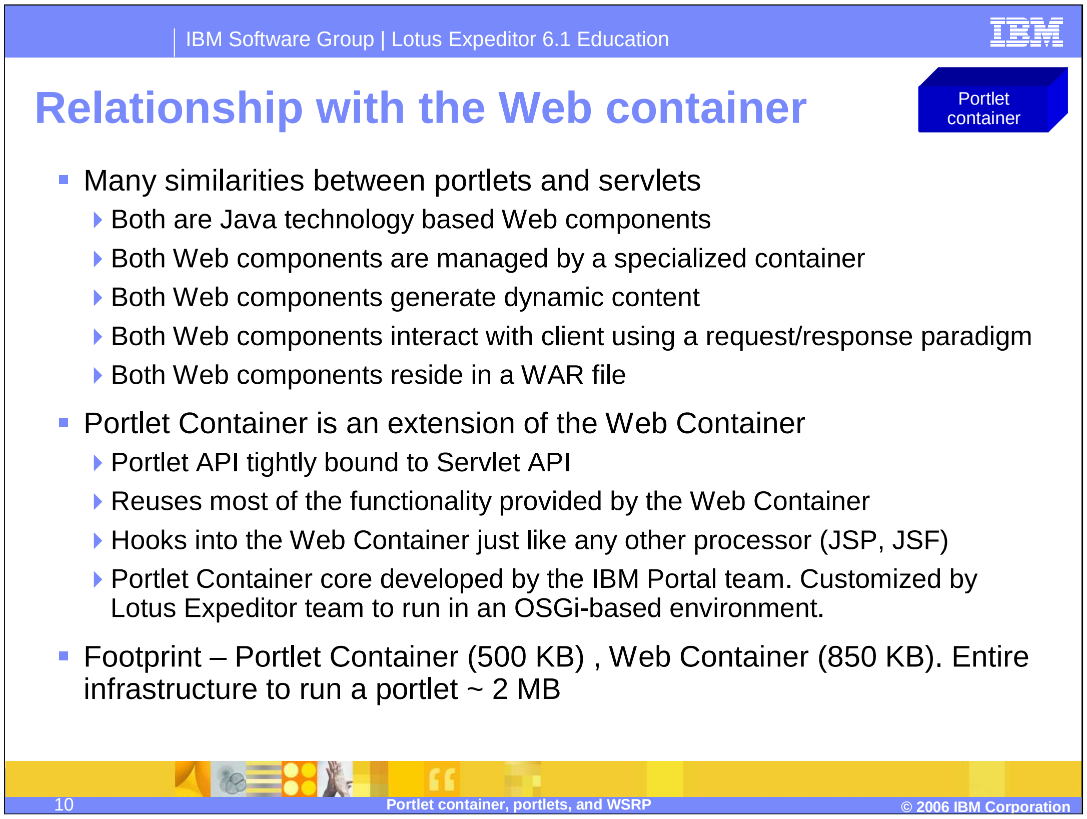
JSR 168 portlet container

Portlet container

- The portlet container provides a runtime environment for JSR 168 portlets
  - JSR 168 portlets can be instantiated, used and finally destroyed
  - portlet applications can be run and accessed either disconnected (offline) or connected (online)
- Portlet Container contains the portlets which are running locally
  - Locally running portlets can use other client services (for example, DB2e and MQe)

The Lotus Expeditor Portlet Container provides a runtime environment for JSR 168 portlets. In the Portlet container, portlets can be instantiated, used and destroyed. Portlet applications can be run and accessed either online or offline.

The Portlet Container supports portlets running locally. Locally running portlets can use other client services, such as, DB2e and MQe.



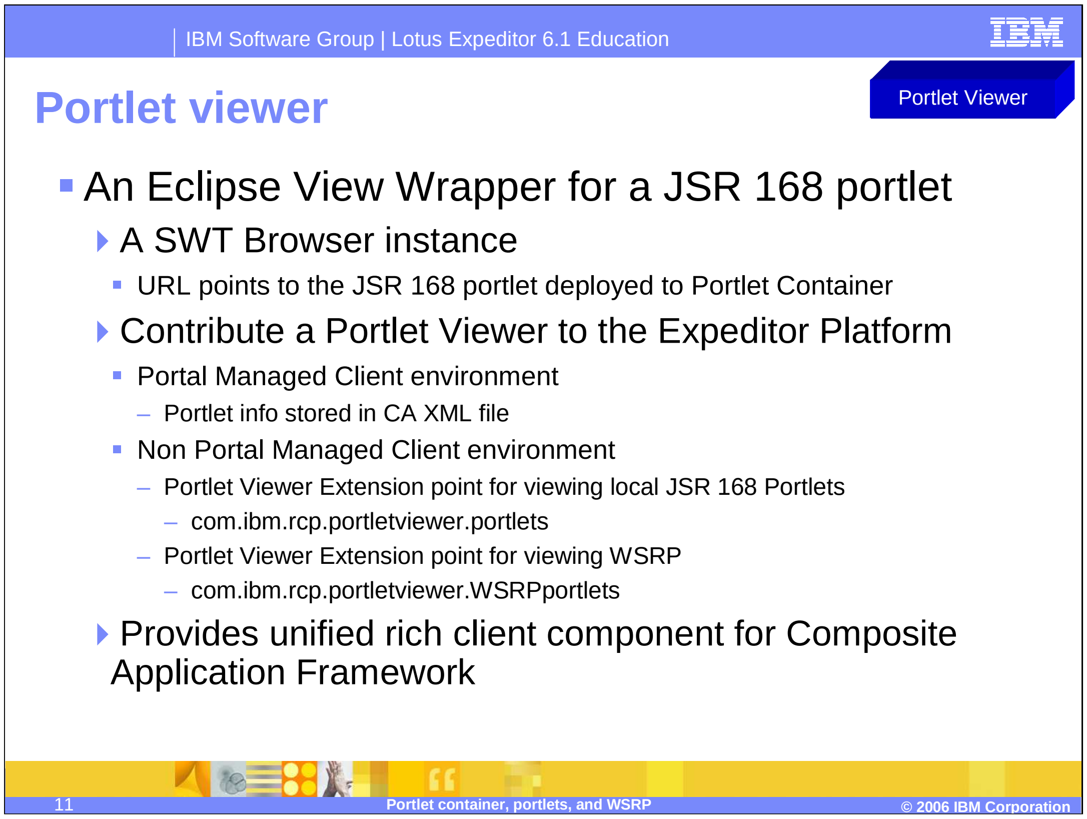
Relationship with the Web container

Portlet container

- Many similarities between portlets and servlets
  - Both are Java technology based Web components
  - Both Web components are managed by a specialized container
  - Both Web components generate dynamic content
  - Both Web components interact with client using a request/response paradigm
  - Both Web components reside in a WAR file
- Portlet Container is an extension of the Web Container
  - Portlet API tightly bound to Servlet API
  - Reuses most of the functionality provided by the Web Container
  - Hooks into the Web Container just like any other processor (JSP, JSF)
  - Portlet Container core developed by the IBM Portal team. Customized by Lotus Expeditor team to run in an OSGi-based environment.
- Footprint – Portlet Container (500 KB) , Web Container (850 KB). Entire infrastructure to run a portlet ~ 2 MB

The Portlet container is an extension of the Web Container. The Portlet API is tightly bound to the Servlet API. There are many similarities between portlets and servlets. For instance, both are managed by a specialized container, both generate dynamic content, both are Web components, both interact with the client using a request/response paradigm and both reside in a WAR file.

The Portlet Container reuses most of the functionality of the Web Container.


IBM Software Group | Lotus Expeditor 6.1 Education

Portlet viewer

Portlet Viewer

- An Eclipse View Wrapper for a JSR 168 portlet
  - A SWT Browser instance
    - URL points to the JSR 168 portlet deployed to Portlet Container
  - Contribute a Portlet Viewer to the Expeditor Platform
    - Portal Managed Client environment
      - Portlet info stored in CA XML file
    - Non Portal Managed Client environment
      - Portlet Viewer Extension point for viewing local JSR 168 Portlets
        - com.ibm.rcp.portletviewer.portlets
      - Portlet Viewer Extension point for viewing WSRP
        - com.ibm.rcp.portletviewer.WSRPportlets
  - Provides unified rich client component for Composite Application Framework

11 Portlet container, portlets, and WSRP © 2006 IBM Corporation

The Portlet Viewer component is an Eclipse View Wrapper for a JSR 168 portlet. It consists of a SWT Browser instance, where the URL points to a JSR 168 portlet that has been deployed in the Portlet Container.

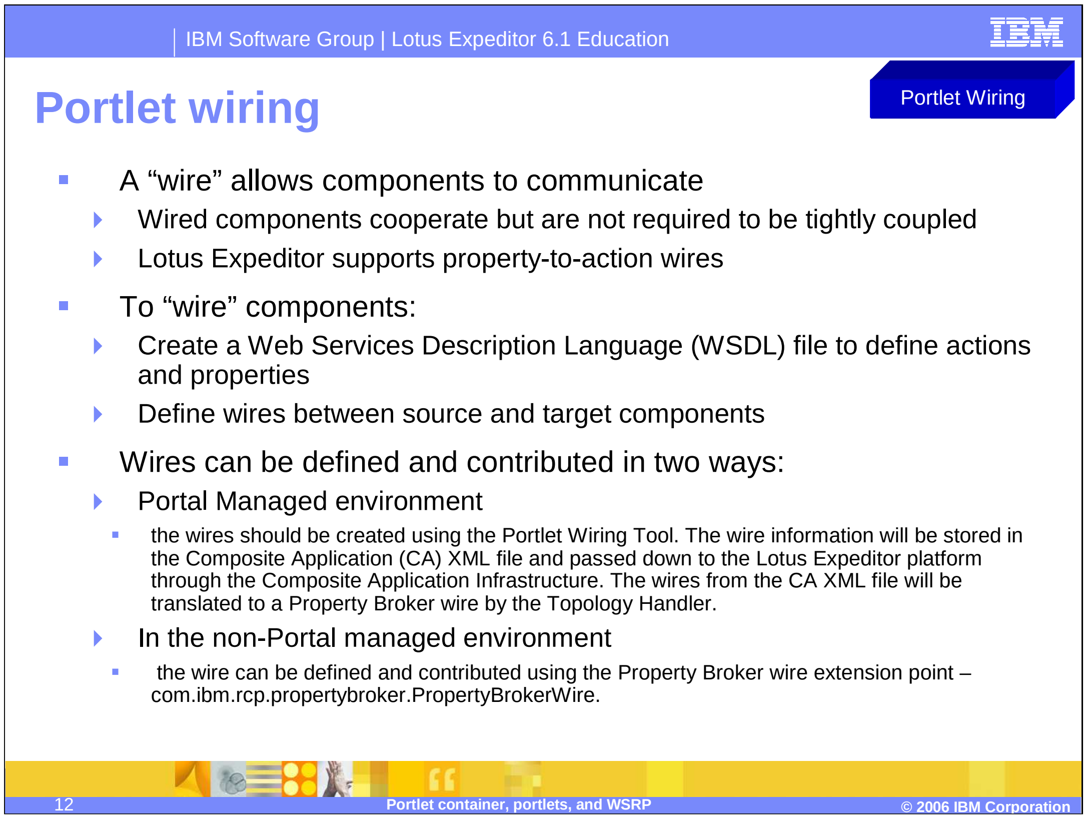
The Portlet Viewer can be used to view local portlets, as well as, WSRP portlets.

A Portlet Viewer instance can be instantiated and contributed to the Lotus Expeditor platform in two ways:

First, in the Portal Managed Client environment, portlet information will be stored in the CA XML file and passed to the Lotus Expeditor platform through the Composite Application Infrastructure.

Second, in the non-Portal Managed Client environment, the Portal Viewer instance can be defined and contributed using the Portlet Viewer extension point. There is an extension point provided for local JSR 168 portlets as well as one for WSRP portlets.

One main benefit of the Portlet Viewer is that it provides a unified rich client component for the composite application framework, for portlet presentation and interaction.

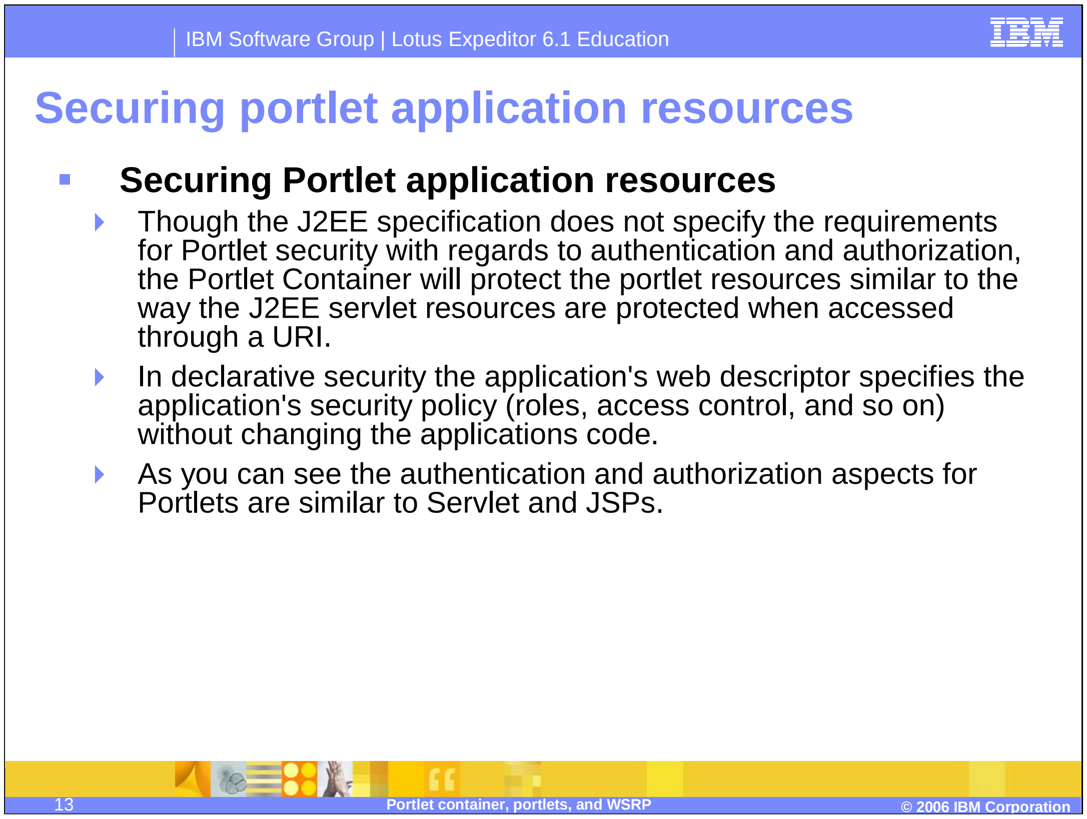


Portlet Wiring

Portlet wiring

- A “wire” allows components to communicate
  - Wired components cooperate but are not required to be tightly coupled
  - Lotus Expeditor supports property-to-action wires
- To “wire” components:
  - Create a Web Services Description Language (WSDL) file to define actions and properties
  - Define wires between source and target components
- Wires can be defined and contributed in two ways:
  - Portal Managed environment
    - the wires should be created using the Portlet Wiring Tool. The wire information will be stored in the Composite Application (CA) XML file and passed down to the Lotus Expeditor platform through the Composite Application Infrastructure. The wires from the CA XML file will be translated to a Property Broker wire by the Topology Handler.
  - In the non-Portal managed environment
    - the wire can be defined and contributed using the Property Broker wire extension point – com.ibm.rcp.propertybroker.PropertyBrokerWire.

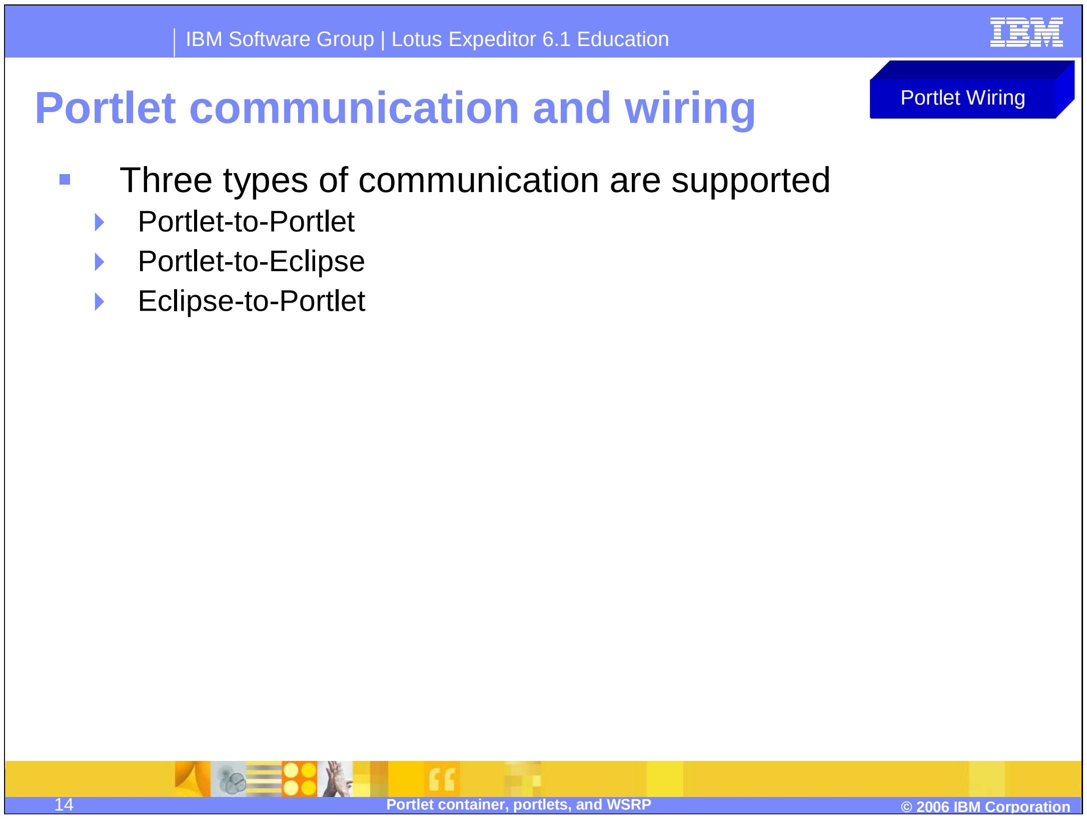
A wire is how components communicate. To wire components, a Web Services Description Language (or WSDL) file must be created to define actions and properties, which defines the wires between source and target components. Lotus Expeditor supports PROPERTY_TO_ACTION wires.



Securing portlet application resources

- **Securing Portlet application resources**
  - Though the J2EE specification does not specify the requirements for Portlet security with regards to authentication and authorization, the Portlet Container will protect the portlet resources similar to the way the J2EE servlet resources are protected when accessed through a URI.
  - In declarative security the application's web descriptor specifies the application's security policy (roles, access control, and so on) without changing the applications code.
  - As you can see the authentication and authorization aspects for Portlets are similar to Servlet and JSPs.

Though the J2EE specification does not specify the requirements for Portlet security with regards to authentication and authorization, the Portlet Container will protect the portlet resources similar to the way the J2EE servlet resources are protected when accessed through a URI. In declarative security, the application's Web descriptor specifies the application's security policy (such as roles and access control) without changing the applications code. As you can see, the authentication and authorization aspects for Portlets are similar to Servlet/JSPs.



IBM

Portlet Wiring

Portlet communication and wiring

- Three types of communication are supported
  - Portlet-to-Portlet
  - Portlet-to-Eclipse
  - Eclipse-to-Portlet

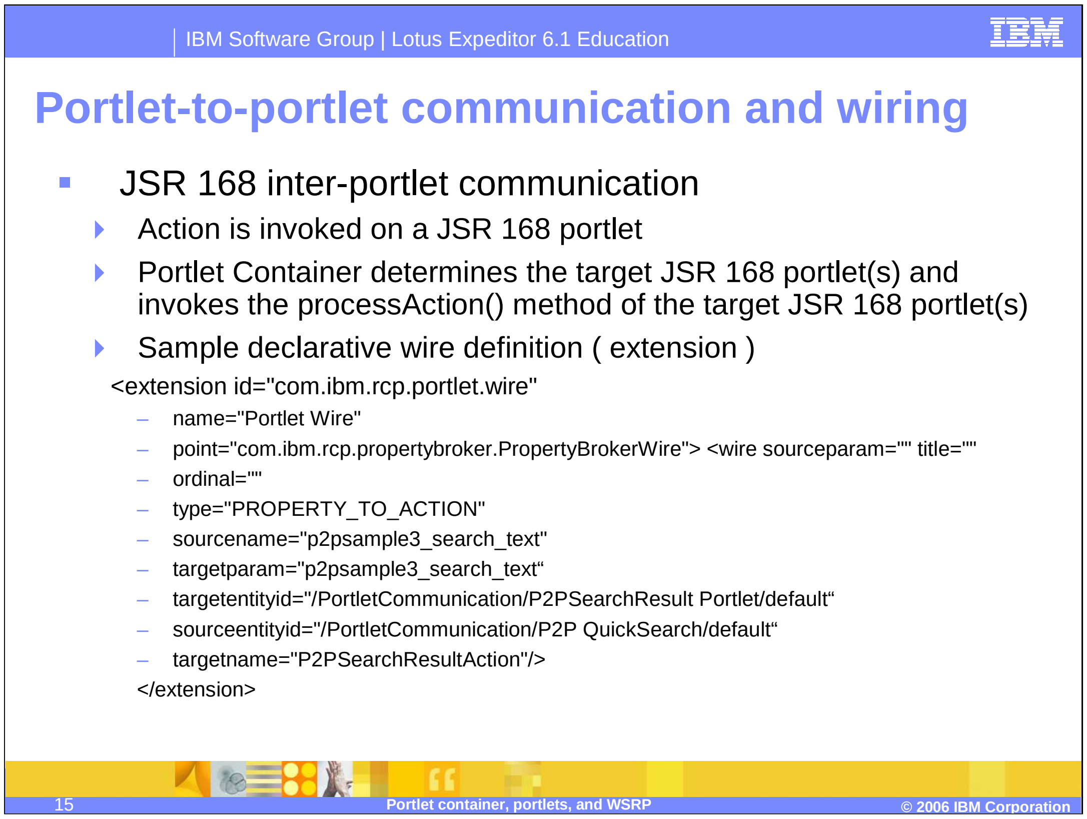
Three types of communication are supported:

•Portlet-to-Portlet

•Portlet-to-Eclipse

•Eclipse-to-Portlet

Next let’s discuss each type.



Portlet-to-portlet communication and wiring

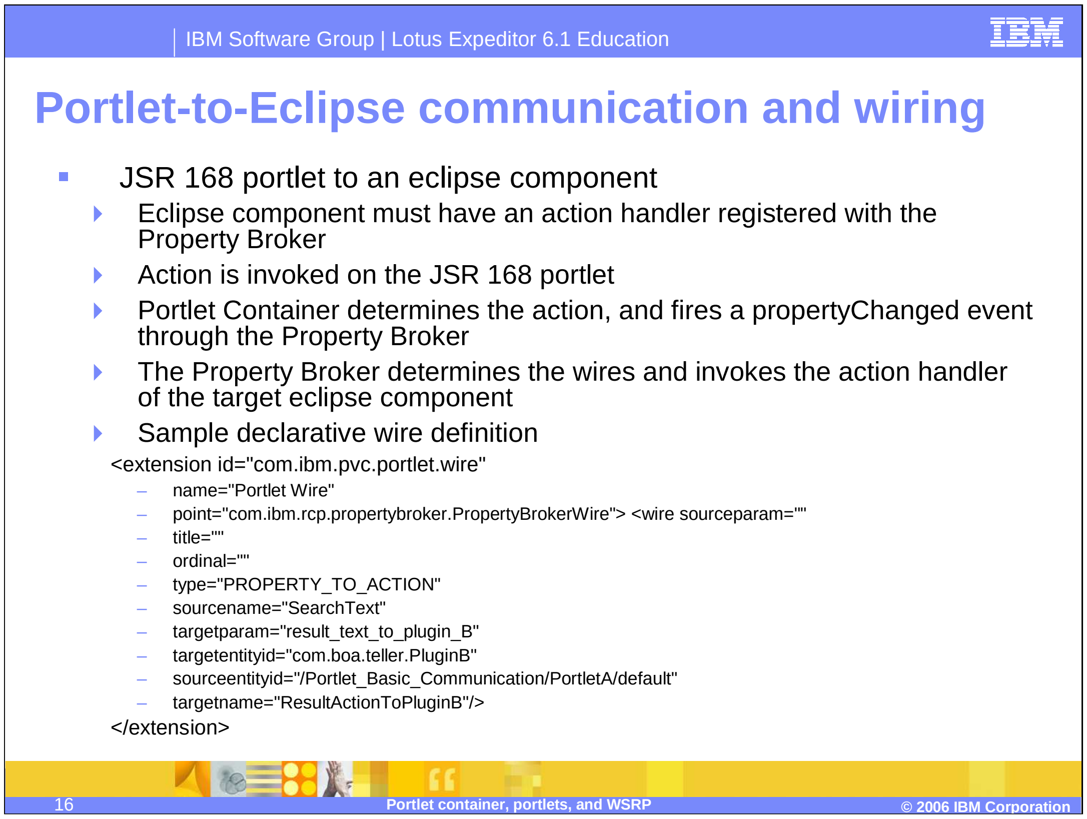
- JSR 168 inter-portlet communication
  - Action is invoked on a JSR 168 portlet
  - Portlet Container determines the target JSR 168 portlet(s) and invokes the processAction() method of the target JSR 168 portlet(s)
  - Sample declarative wire definition (extension)

<extension id="com.ibm.rcp.portlet.wire"

- name="Portlet Wire"
- point="com.ibm.rcp.propertybroker.PropertyBrokerWire"> <wire sourceparam="" title=""
- ordinal=""
- type="PROPERTY_TO_ACTION"
- sourcename="p2psample3_search_text"
- targetparam="p2psample3_search_text“
- targetentityid="/PortletCommunication/P2PSearchResult Portlet/default“
- sourceentityid="/PortletCommunication/P2P QuickSearch/default“
- targetname="P2PSearchResultAction"/>

</extension>

This slide shows a sample declarative wire definition for Portlet-to-Portlet communication and wiring.

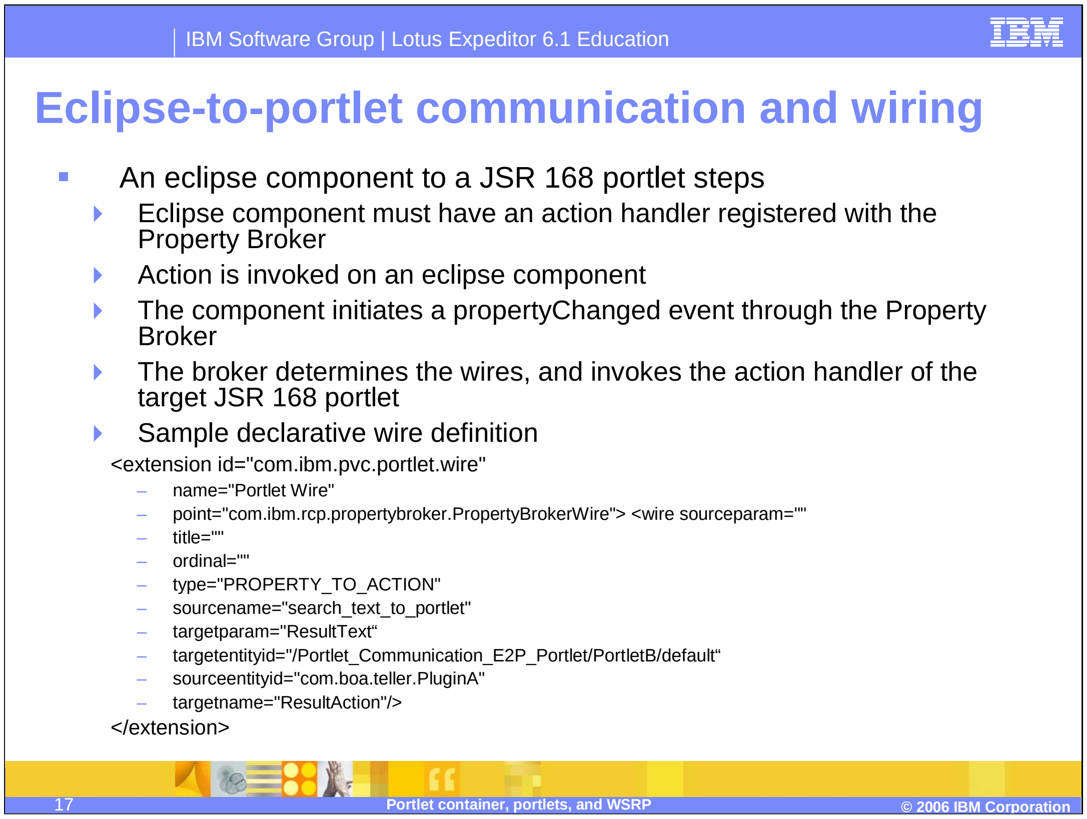


Portlet-to-Eclipse communication and wiring

- JSR 168 portlet to an eclipse component
  - Eclipse component must have an action handler registered with the Property Broker
  - Action is invoked on the JSR 168 portlet
  - Portlet Container determines the action, and fires a propertyChanged event through the Property Broker
  - The Property Broker determines the wires and invokes the action handler of the target eclipse component
  - Sample declarative wire definition

```
<extension id="com.ibm.pvc.portlet.wire"
  – name="Portlet Wire"
  – point="com.ibm.rcp.propertybroker.PropertyBrokerWire"> <wire sourceparam=""
  – title=""
  – ordinal=""
  – type="PROPERTY_TO_ACTION"
  – sourcename="SearchText"
  – targetparam="result_text_to_plugin_B"
  – targetentityid="com.boa.teller.PluginB"
  – sourceentityid="/Portlet_Basic_Communication/PortletA/default"
  – targetname="ResultActionToPluginB"/>
</extension>
```

This slide shows a sample declarative wire definition for Portlet-to-Eclipse communication and wiring.



Eclipse-to-portlet communication and wiring

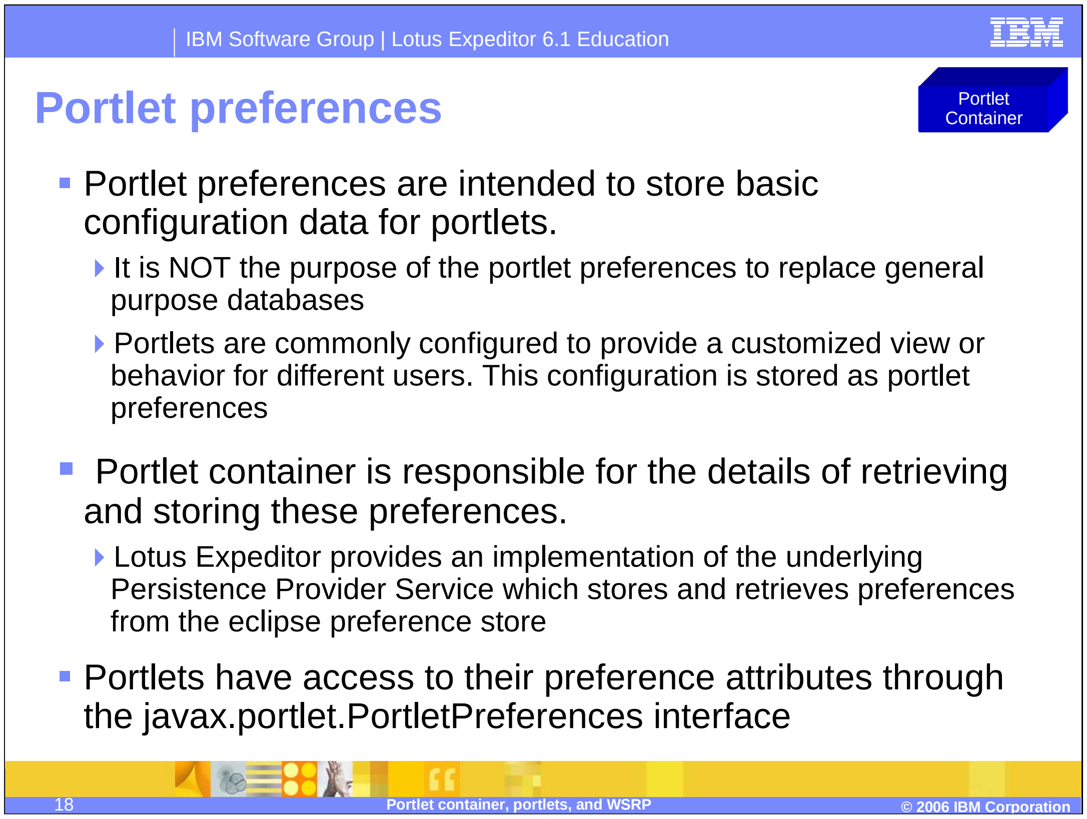
- An eclipse component to a JSR 168 portlet steps
  - Eclipse component must have an action handler registered with the Property Broker
  - Action is invoked on an eclipse component
  - The component initiates a propertyChanged event through the Property Broker
  - The broker determines the wires, and invokes the action handler of the target JSR 168 portlet
  - Sample declarative wire definition

<extension id="com.ibm.pvc.portlet.wire"

- name="Portlet Wire"
- point="com.ibm.rcp.propertybroker.PropertyBrokerWire"> <wire sourceparam=""
- title=""
- ordinal=""
- type="PROPERTY_TO_ACTION"
- sourcename="search_text_to_portlet"
- targetparam="ResultText“
- targetentityid="/Portlet_Communication_E2P_Portlet/PortletB/default“
- sourceentityid="com.boa.teller.PluginA"
- targetname="ResultAction"/>

</extension>

This slide shows a sample declarative wire definition for Eclipse-to-Portlet communication and wiring.



Portlet preferences

Portlet Container

- Portlet preferences are intended to store basic configuration data for portlets.
  - It is NOT the purpose of the portlet preferences to replace general purpose databases
  - Portlets are commonly configured to provide a customized view or behavior for different users. This configuration is stored as portlet preferences
- Portlet container is responsible for the details of retrieving and storing these preferences.
  - Lotus Expeditor provides an implementation of the underlying Persistence Provider Service which stores and retrieves preferences from the eclipse preference store
- Portlets have access to their preference attributes through the javax.portlet.PortletPreferences interface

Portlet preferences are intended to store basic configuration data for portlets.

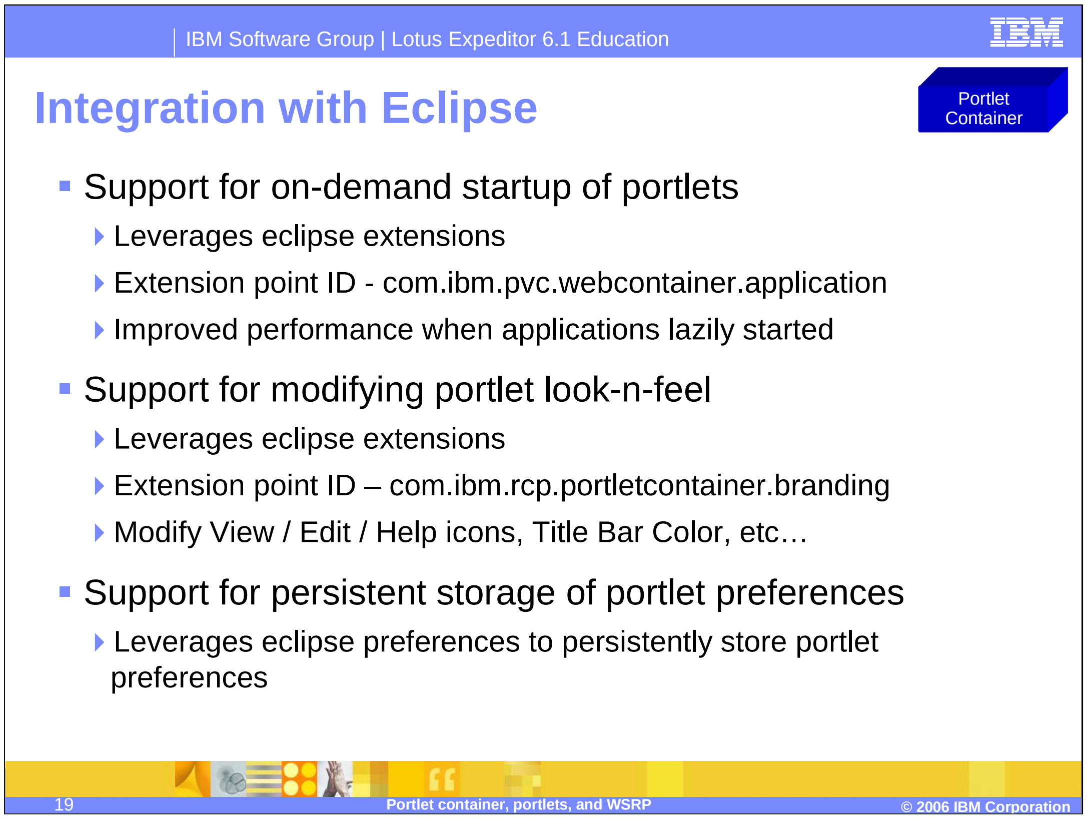
•It is NOT the purpose of the portlet preferences to replace general purpose databases

•Portlets are commonly configured to provide a customized view or behavior for different users. This configuration is stored as portlet preferences.

The Portlet Container is responsible for the details of retrieving and storing these preferences.

•Lotus Expeditor provides an implementation of the underlying Persistence Provider Service which stores and retrieves preferences from the eclipse preference store.

Portlets have access to their preference attributes through the javax.portlet.PortletPreferences interface.

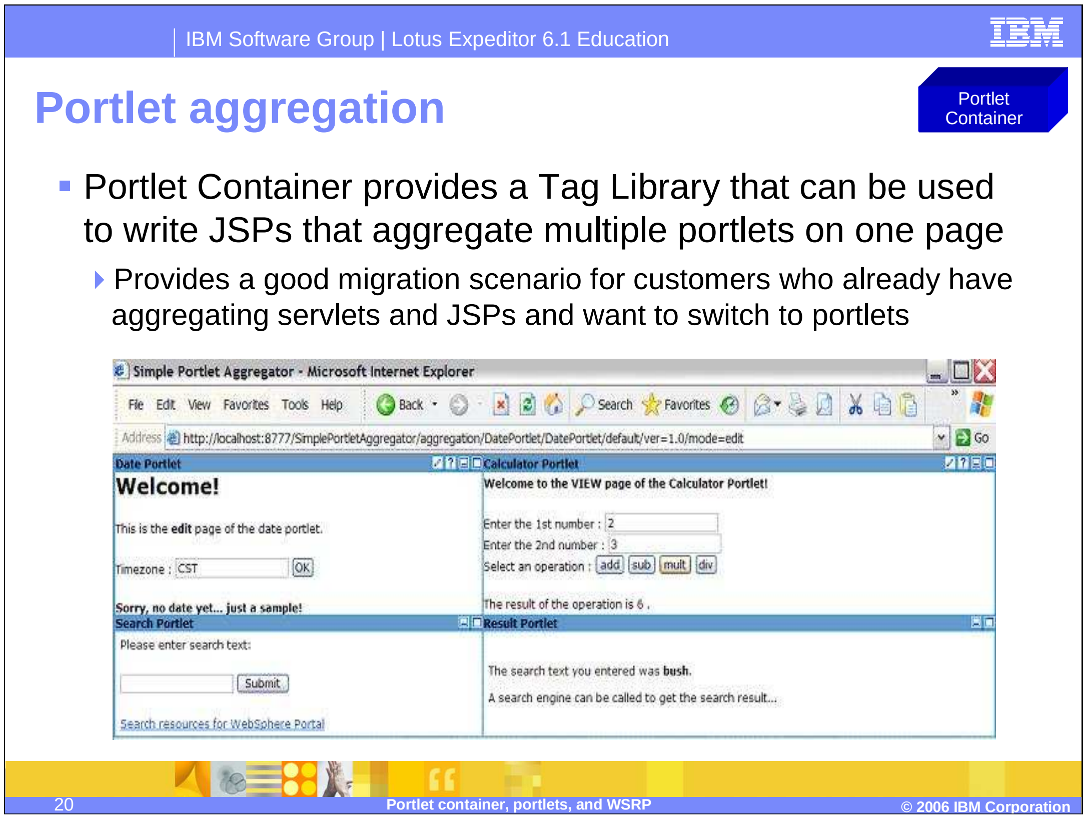


Integration with Eclipse

Portlet Container

- Support for on-demand startup of portlets
  - Leverages eclipse extensions
  - Extension point ID - com.ibm.pvc.webcontainer.application
  - Improved performance when applications lazily started
- Support for modifying portlet look-n-feel
  - Leverages eclipse extensions
  - Extension point ID – com.ibm.rcp.portletcontainer.branding
  - Modify View / Edit / Help icons, Title Bar Color, etc…
- Support for persistent storage of portlet preferences
  - Leverages eclipse preferences to persistently store portlet preferences

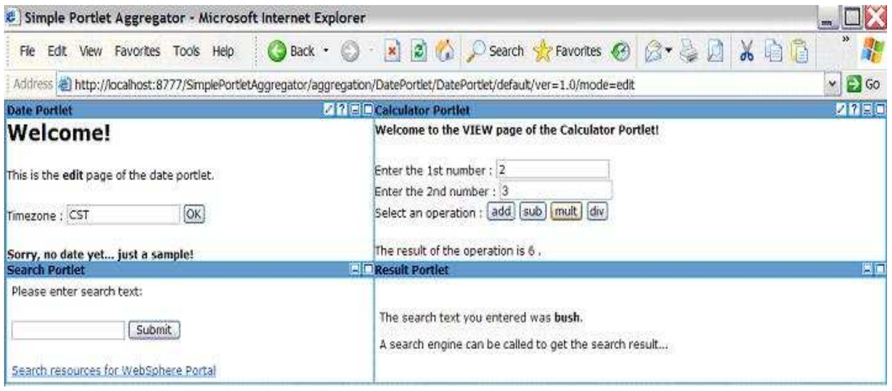
The Portlet Container leverages Eclipse features, including support for on-demand startup, branding and preferences of portlets. For on-demand startup, the webcontainer application extension point is provided. For branding, use the portlet container branding extension point. Branding provides support to modify icons, such as view, edit and help icons, as well as Title Bar Color. For persistent storage of portlet preferences, use eclipse preferences.



Portlet aggregation

Portlet Container

- Portlet Container provides a Tag Library that can be used to write JSPs that aggregate multiple portlets on one page
  - Provides a good migration scenario for customers who already have aggregating servlets and JSPs and want to switch to portlets



http://localhost:8777/SimplePortletAggregator/aggregation/DatePortlet/DatePortlet/default/ver=1.0/mode=edit

Welcome!

This is the **edit** page of the date portlet.

Timezone : CST

Sorry, no date yet... just a sample!

Welcome to the VIEW page of the Calculator Portlet!

Enter the 1st number : 2

Enter the 2nd number : 3

Select an operation :

The result of the operation is 6 .

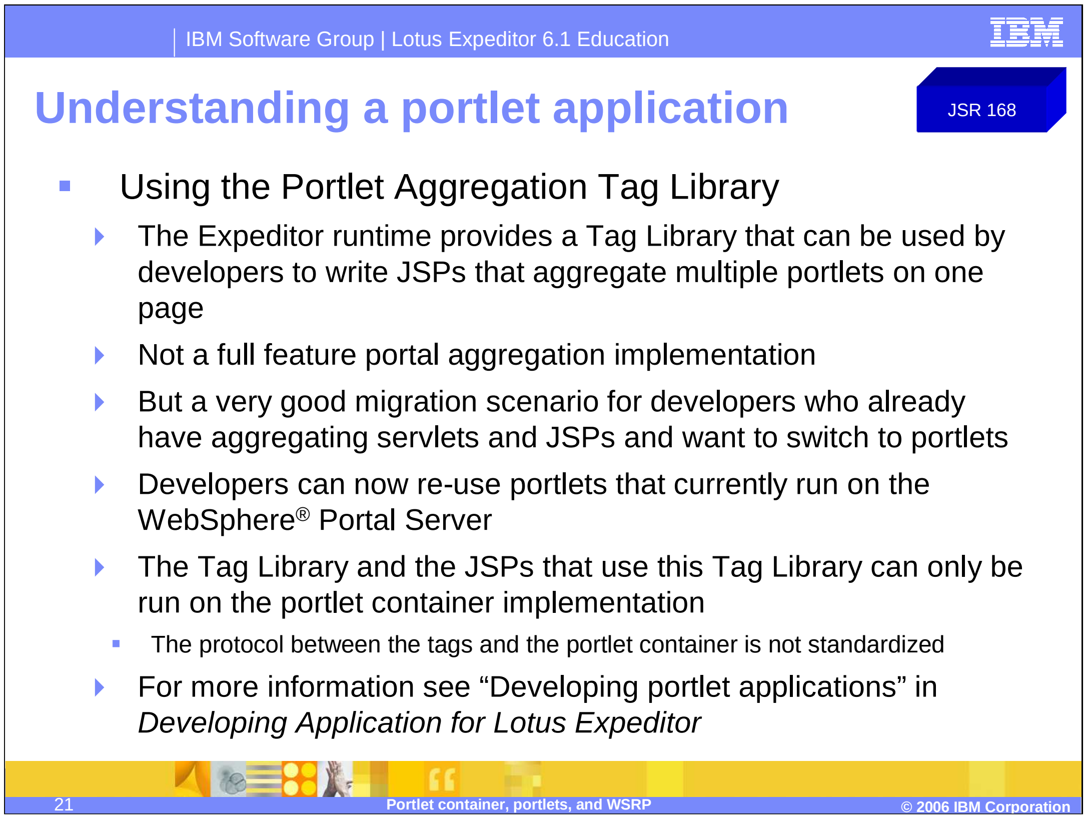
Please enter search text:

Search resources for WebSphere Portal

The search text you entered was **bush**.

A search engine can be called to get the search result...

The Portlet Container provides a Tag Library that can be used to write JSPs that aggregate multiple portlets on one page. This provides a good migration scenario for customers who already have aggregating servlets and JSPs and want to switch to portlets.



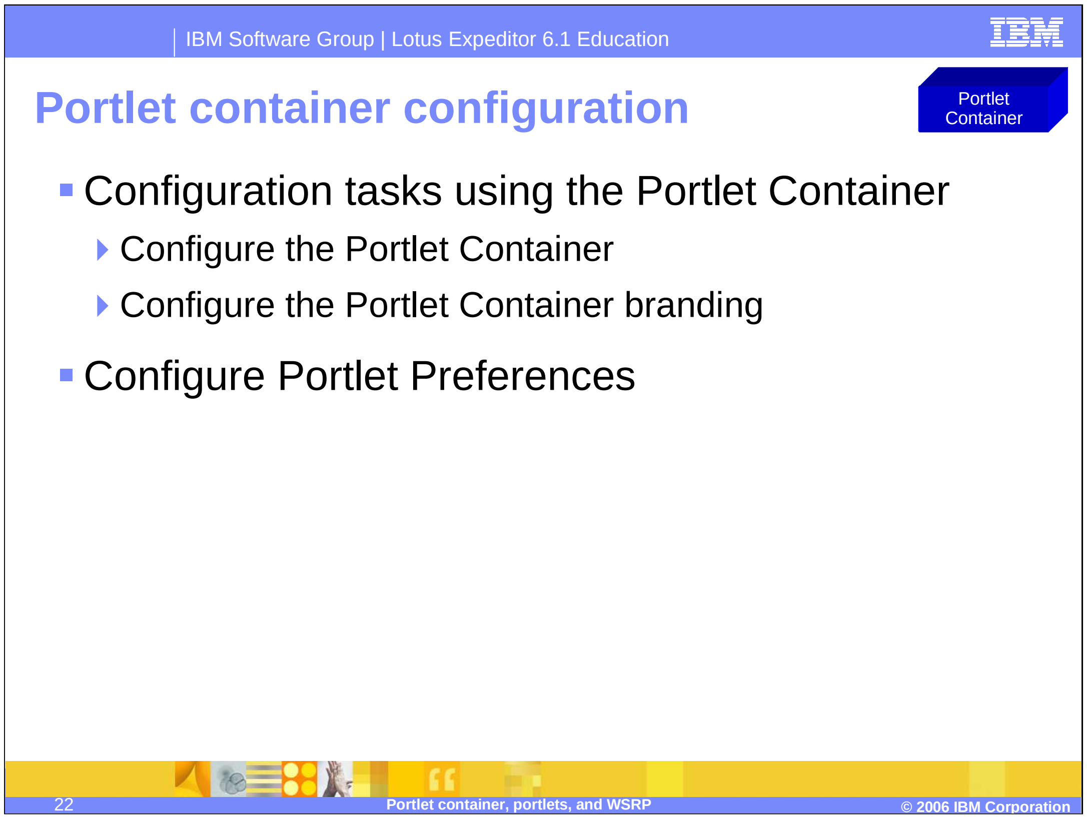
Understanding a portlet application

JSR 168

- Using the Portlet Aggregation Tag Library
  - The Expeditor runtime provides a Tag Library that can be used by developers to write JSPs that aggregate multiple portlets on one page
  - Not a full feature portal aggregation implementation
  - But a very good migration scenario for developers who already have aggregating servlets and JSPs and want to switch to portlets
  - Developers can now re-use portlets that currently run on the WebSphere® Portal Server
  - The Tag Library and the JSPs that use this Tag Library can only be run on the portlet container implementation
    - The protocol between the tags and the portlet container is not standardized
  - For more information see “Developing portlet applications” in *Developing Application for Lotus Expeditor*

The Lotus Expeditor 6.1 runtime provides a Tag Library that can be used by developers to write JSPs that aggregate multiple portlets on one page. This capability will not provide the developer with a full feature portal aggregation implementation, but provide a good migration scenario for developers who already have aggregating servlets and JSPs and want to switch to portlets.

The biggest advantage for developers is that they can re-use portlets that currently run on the WebSphere Portal Server on the Lotus Expeditor 6.1 client runtime. However, the Tag Library and the JSPs that use this Tag Library can only be run on the WebSphere portlet container implementation. The protocol between the tags and the portlet container is not standardized.

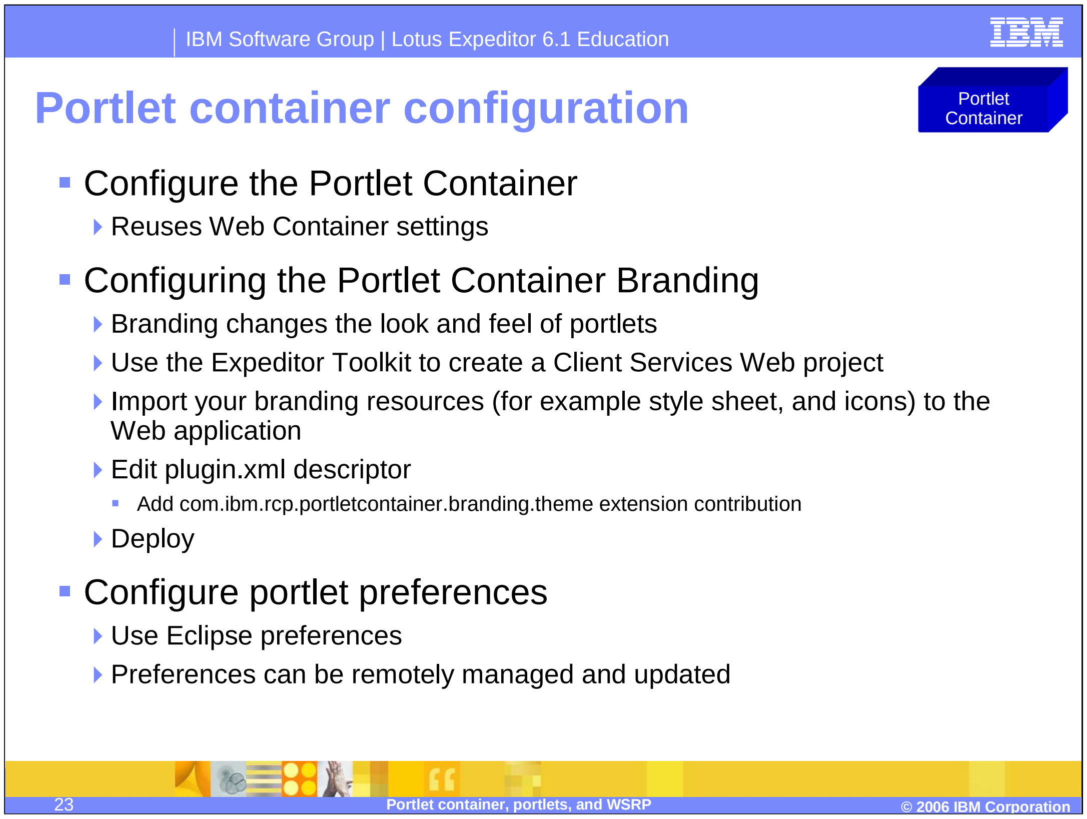


Portlet container configuration

Portlet Container

- Configuration tasks using the Portlet Container
  - Configure the Portlet Container
  - Configure the Portlet Container branding
- Configure Portlet Preferences

The Portlet Container can be configured using Java System properties or using Configuration Admin. Configurable properties include the port to listen on and the IP address to listen on. A full list of configurable properties are documented in the *Assembling and Deploying Lotus Expeditor Applications* guide and the *Developing Application for Lotus Expeditor* guide.



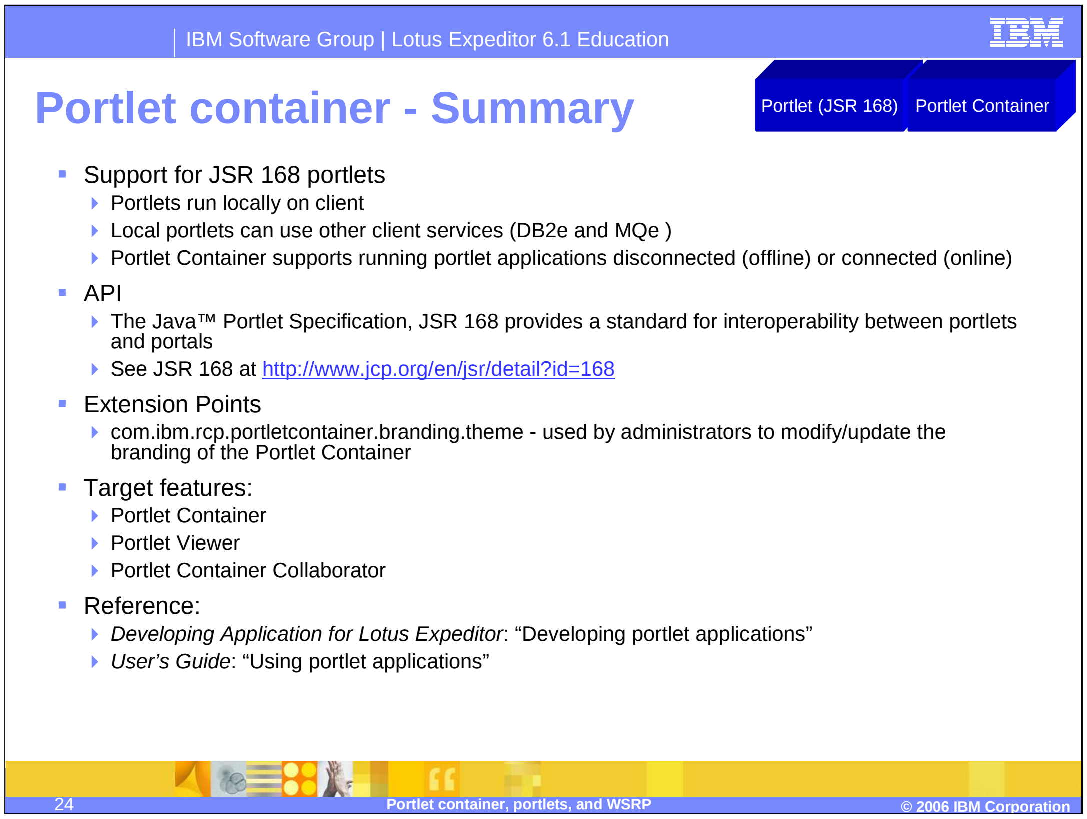
Portlet container configuration

Portlet Container

- Configure the Portlet Container
  - Reuses Web Container settings
- Configuring the Portlet Container Branding
  - Branding changes the look and feel of portlets
  - Use the Expeditor Toolkit to create a Client Services Web project
  - Import your branding resources (for example style sheet, and icons) to the Web application
  - Edit plugin.xml descriptor
    - Add com.ibm.rcp.portletcontainer.branding.theme extension contribution
  - Deploy
- Configure portlet preferences
  - Use Eclipse preferences
  - Preferences can be remotely managed and updated

The Portlet Container is an additional extension processor which leverages Web Container functionality and reuses the Web Container settings (server address and server port, for example).

Use Eclipse Preferences to persistently store modified and updated portlet preferences. The default set of portlet preferences can be remotely managed and updated.

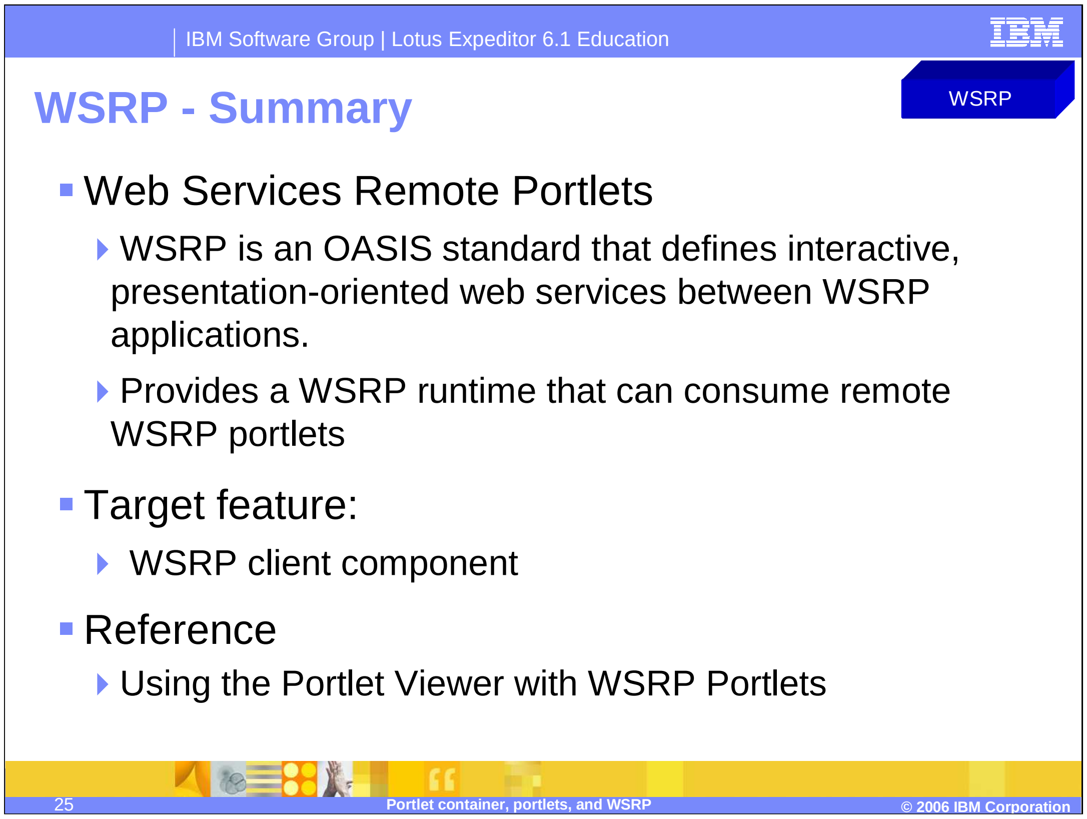


Portlet container - Summary

Portlet (JSR 168)
Portlet Container

- Support for JSR 168 portlets
  - Portlets run locally on client
  - Local portlets can use other client services (DB2e and MQe)
  - Portlet Container supports running portlet applications disconnected (offline) or connected (online)
- API
  - The Java™ Portlet Specification, JSR 168 provides a standard for interoperability between portlets and portals
  - See JSR 168 at http://www.jcp.org/en/jsr/detail?id=168
- Extension Points
  - com.ibm.rcp.portletcontainer.branding.theme - used by administrators to modify/update the branding of the Portlet Container
- Target features:
  - Portlet Container
  - Portlet Viewer
  - Portlet Container Collaborator
- Reference:
  - *Developing Application for Lotus Expeditor*: “Developing portlet applications”
  - *User’s Guide*: “Using portlet applications”

In summary, the Portlet Container provides a runtime environment for portlets so portlets can be instantiated, used and finally destroyed. The Portlet Container leverages existing standard-based implementations (OSGi and JSR 168) and provides a runtime environment in which users and developers can access and run JSR 168 portlets that work across compliant portlet containers. The Portlet Container allows users and developers alike to access and run portlet applications either disconnected (that is, offline) or connected (or online).



WSRP - Summary

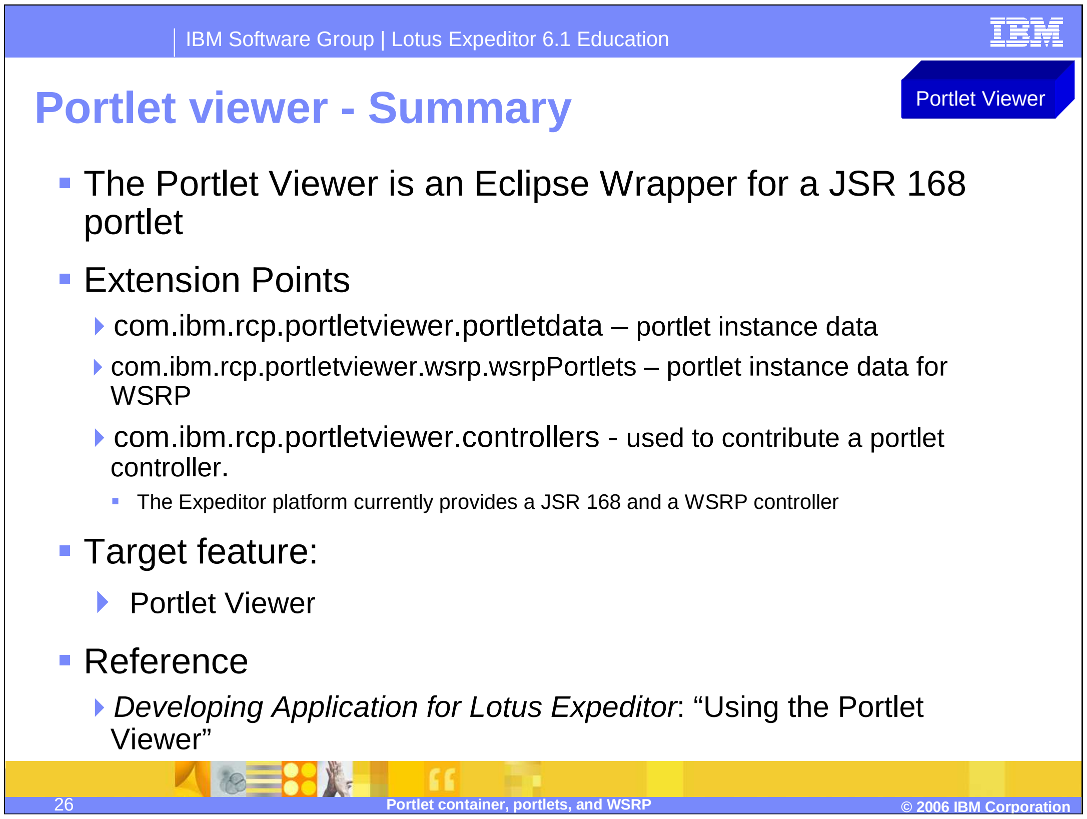
WSRP

- Web Services Remote Portlets
  - WSRP is an OASIS standard that defines interactive, presentation-oriented web services between WSRP applications.
  - Provides a WSRP runtime that can consume remote WSRP portlets
- Target feature:
  - WSRP client component
- Reference
  - Using the Portlet Viewer with WSRP Portlets

Expeditor Client provides the WSRP runtime feature.

Through the WSRP runtime, the portlet viewer is used to support presentation and interaction with remote portlets inside a portal server.

WSRP is an OASIS standard that defines interactive, presentation-oriented Web services between WSRP-enabled applications, such as the Portal server or the WSRP runtime within the Expeditor Client. The WSRP runtime retrieves portlet content from remote portlets and processes user interaction.

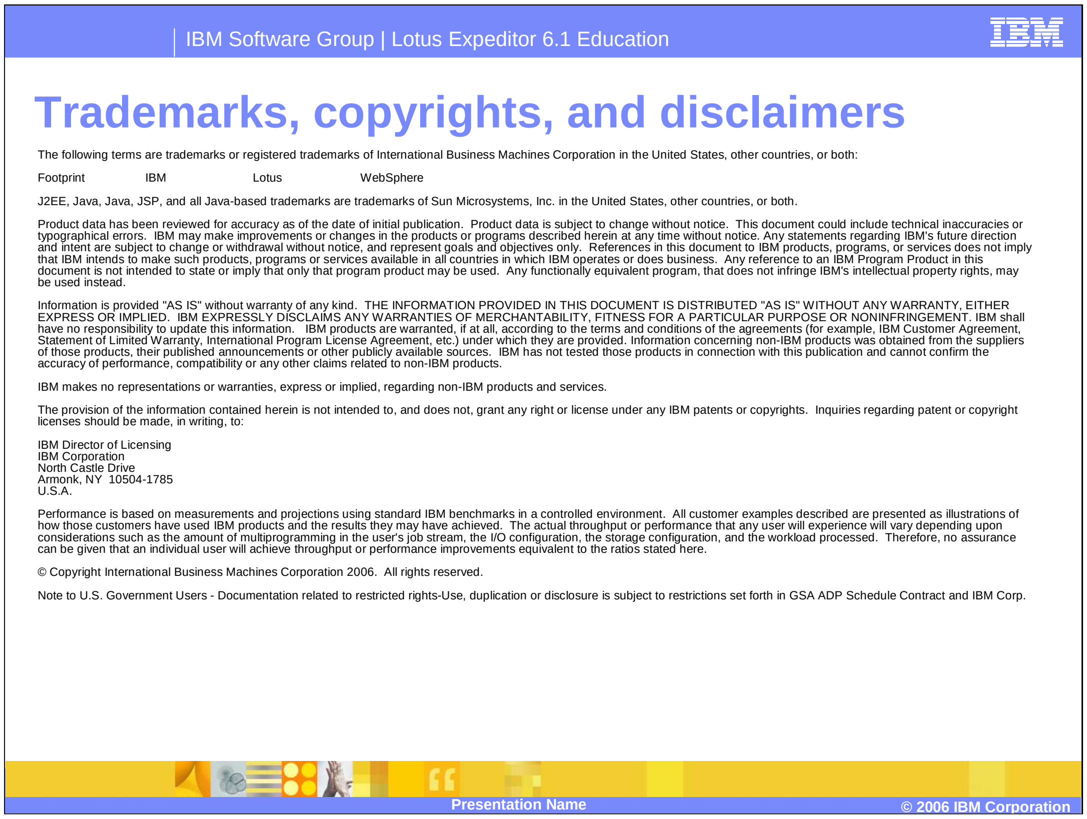


Portlet Viewer

Portlet viewer - Summary

- The Portlet Viewer is an Eclipse Wrapper for a JSR 168 portlet
- Extension Points
  - com.ibm.rcp.portletviewer.portletdata – portlet instance data
  - com.ibm.rcp.portletviewer.wsrp.wsrpPortlets – portlet instance data for WSRP
  - com.ibm.rcp.portletviewer.controllers - used to contribute a portlet controller.
    - The Expeditor platform currently provides a JSR 168 and a WSRP controller
- Target feature:
  - Portlet Viewer
- Reference
  - *Developing Application for Lotus Expeditor*: “Using the Portlet Viewer”

The Portlet Viewer is an Eclipse view wrapper for a JSR 168 portlet. It consists of a SWT browser instance whose URL points to a JSR 168 portlet deployed on the Portlet Container. The main benefit of the Portlet Viewer is that it provides a unified rich client component for the composite application framework, for portlet presentation and interaction.



Trademarks, copyrights, and disclaimers

This concludes the presentation.