WebSphere software

IBM

e business software

Portlet API and Portlet Migration

WebSphere Portal 4.1

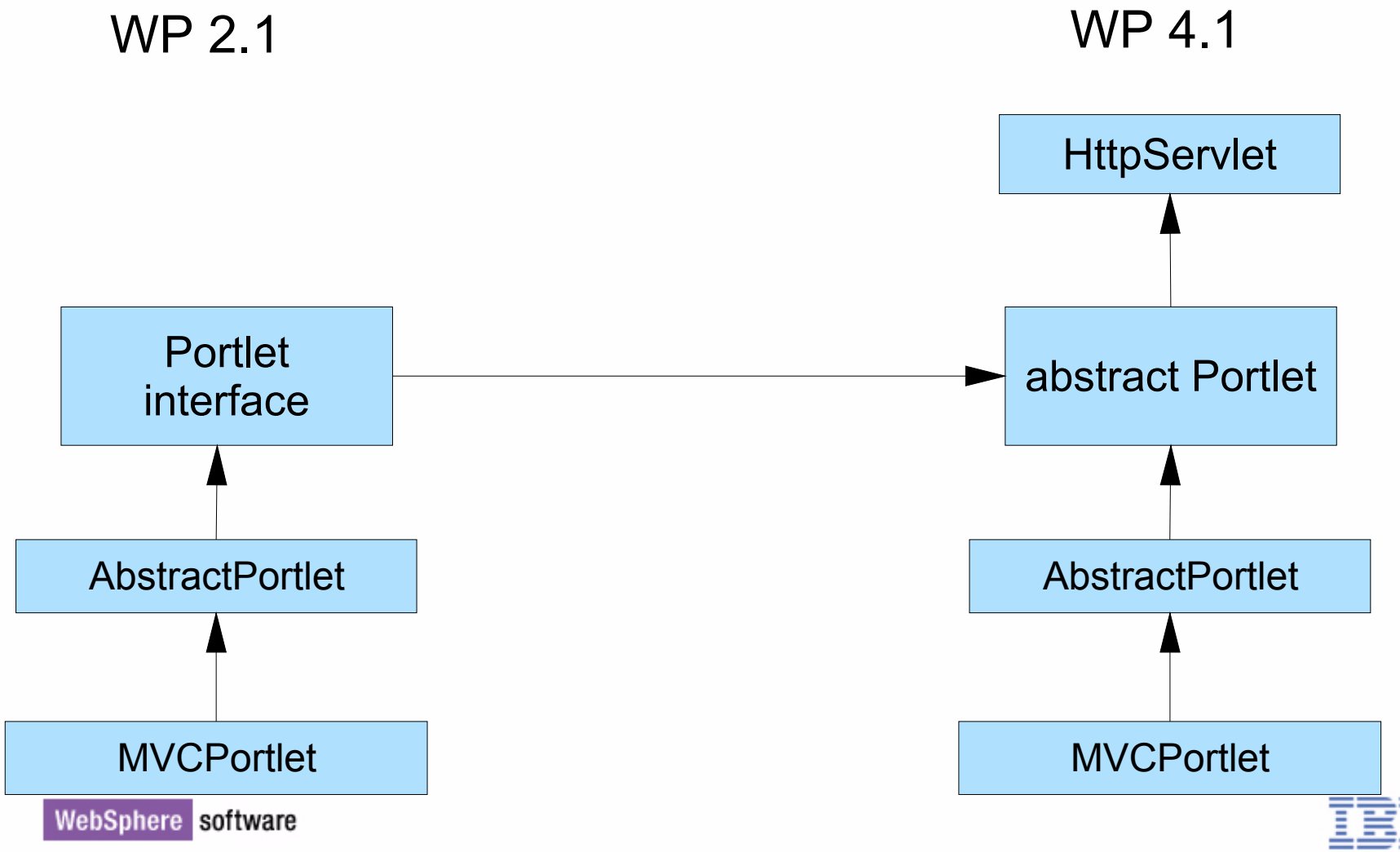
IBM Software Group

Agenda

- Portlet API
- Migration Considerations
- Portlet Migration Tool
- Other Migration Tools
- Developing Portlets to run on 2.1 and 4.1

Portlet API Inheritance Changes

- Portlet API now inherit directly from HttpServlet. It is no longer an independent entity.

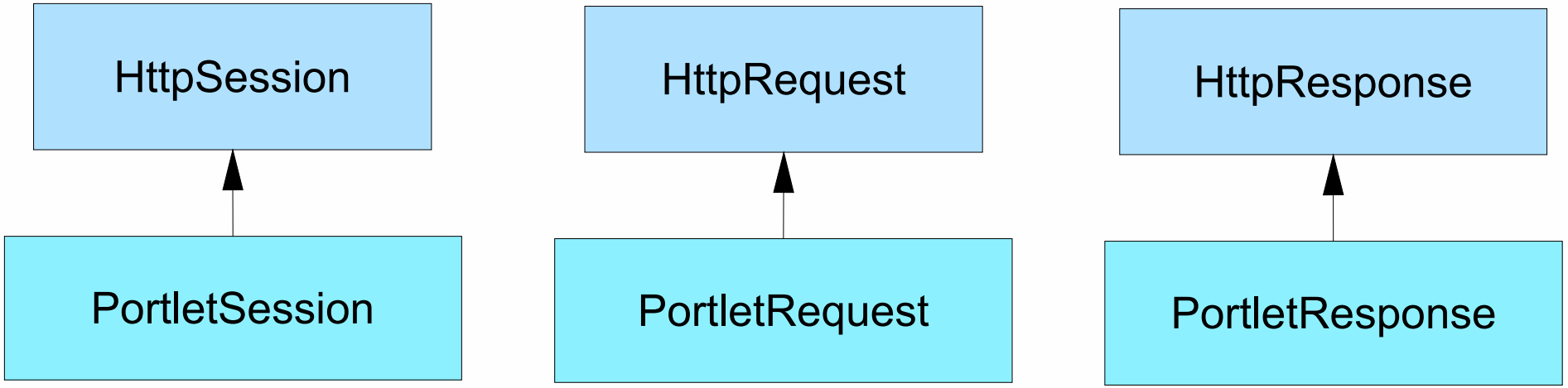


WebSphere software

IBM

Portlet Configuration Inheritance Changes

- Portlet configuration objects now inherit from their corresponding HTTP counterpart. They are no longer independent objects.
- Example on next page.



Casting Request, Response and Session Objects

- In WebSphere Portal v2.1:
  - HttpServletRequest httpRequest = ((PortletRequestImpl) request).getServletRequest();
  - PortletSession myPortletSession = httpRequest.getSession();
  - String myServletID = myPortletSession.getID(); // HttpSession.getID()
- In WebSphere Portal v4.1:
  - PortletSession myPortletSession = request.getSession();
  - String myServletID = myPortletSession.getID(); // HttpSession.getID()

Portal Object	Server Object
PortletSession	HttpSession
PortletRequest	HttpRequest
PortletResponse	HttpResponse

Portlet API 1.1 : Java Class Changes

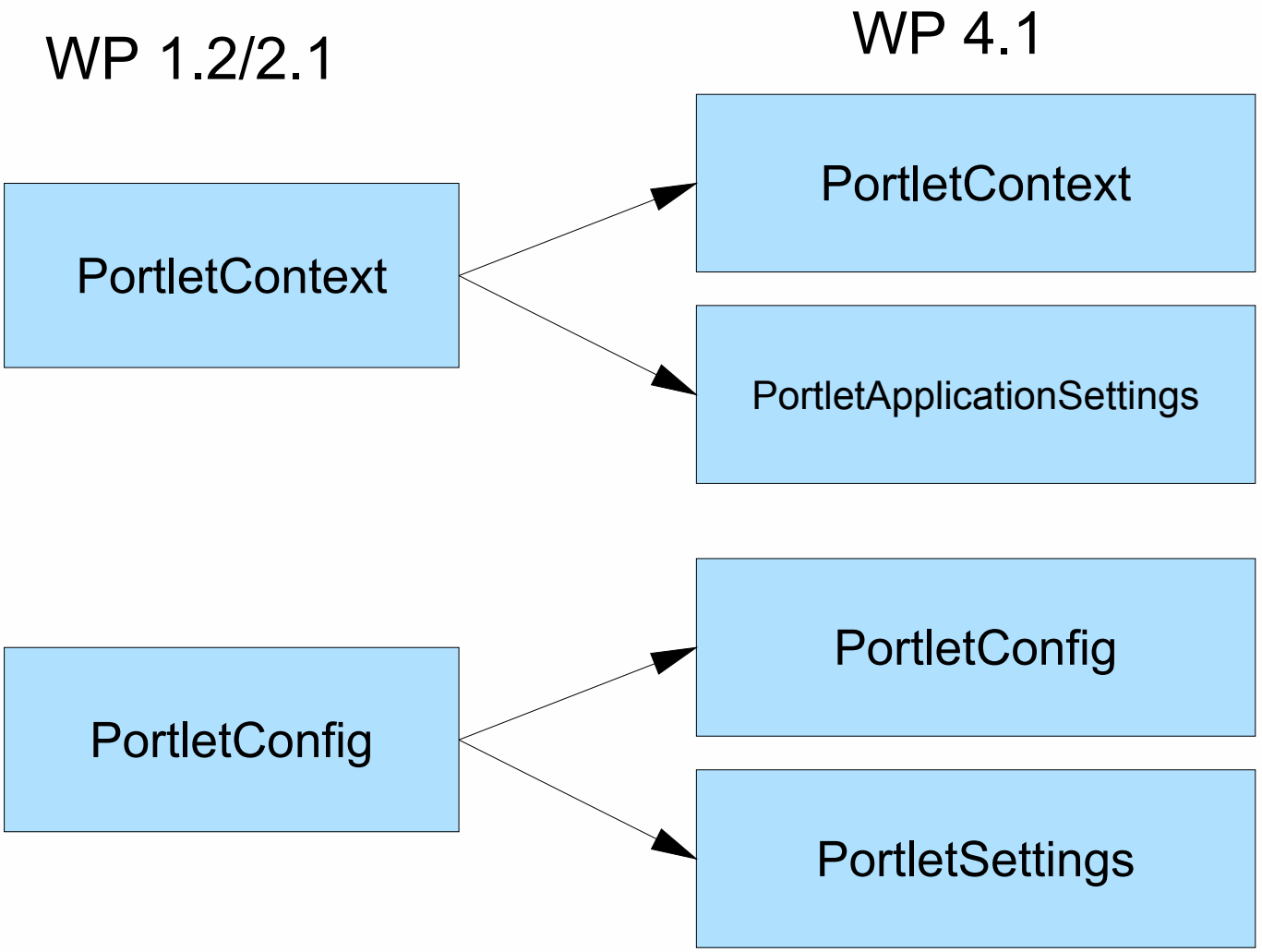
Class	Method	Change	Comments
Event	GetPortlet()	Deprecated	See Event Javadoc
PortletConfig	getAttribute()	Changed method name	Use getInitParameter()
PortletContext	Send	Deprecated	See PortletContext Javadoc for new method signatures
PortletData	removeAllAttributes()	Deprecated	No replacement
PortletRequest	getUser()	Added method	
PortletRequest	getPortletSettings()	Added method	PortletSettings are for configuration data for the portlet. PortletData is still the place to store user settings and values.
PortletRequest	getSession()	Changed method name	Now getPortletSession()
PortletResponse	encodeURI()	Deprecated	Use encodeURL()
PortletResponse	getCharacterSet()	Deprecated	Use getCharacterEncoding()
PortletSession	GetUser()	Deprecated	Use portletRequest.getUser()
PortletApplicationSettings		New Class	Manages portlet application settings that are updated by the portal administrator
PortletSettings		New Class	Manages portlet settings that are updated by the portal administrator

Java Server Pages (JSP) Changes

- In previous releases of the portal the built-in JSP variables request and response were of type PortletRequest and PortletResponse.
- In WebSphere Portal Version 4.1 the class of these variables are HttpServletRequest and HttpServletResponse.
- An exception is thrown if you attempt to cast these variables to PortletRequest and PortletResponse.
- If you need access to PortletRequest or PortletResponse specific methods, add the following code to the beginning of your JSP:
  -
  - <%@ taglib uri="/WEB-INF/tld/portlet.tld" prefix="portletAPI" %> <portletAPI:init />
  -
- A call to the init tag will create three variables properly typed and initialized: PortletRequest, PortletResponse, and PortletSession.

Portlet API Entity Changes

- New Objects for the Management of Configuration Data



WebSphere software

IBM

Other Portlet API Changes

- Portlet Interface is removed
  - Portlets extend AbstractPortlet, MVCPortlet or PortletAdapter
- Portlet messaging has been enhanced. Portlets may now send messages to portlets in other portlet applications using the DefaultPortletMessage class
  - Both the message sending Portlet and the message receiving portlet must still reside on the same portal page in order for messages to be delivered
- Portlets may store configuration values in the PortletData object only when the portlet is in edit mode. Previously this function was available in the portlet view mode, and during all portlet event handling
- Class SaxPortlet has been removed and is no longer supported

Other Portlet API Changes

- Retrieval of JAAS user subject has changed
  - In WPS 2.1 access security information by obtaining a reference to the user's subject

```
PortletSession session = portletRequest.getSession();  // Get the Portlet Session
org.apache.jetspeed.portletcontainer.UserImpl  user =(org.apache.jetspeed.portletcontainer.UserImpl)
session.getUser();  // Get the User Object
Subject subject = user.getSubject();                // Next, grab the Subject
```

  - In WP 4.1 User subject is accessed from the portlet context

```
import org.apache.jetspeed.portlet.*;
import com.ibm.wps.portletservice.credentialvault.CredentialVaultService;
import javax.security.auth.Subject;
PortletContext        context = this.getPortletConfig().getContext();
CredentialVaultService service = (CredentialVaultService) context.getService(CredentialVaultService.class);
Subject userSubject = service.getUserSubject(portletRequest);
```

- Retrieving a Reference to the User Object
  - User myPortletUser = portletRequest.getUser():
- Packaging
  - Portlet are now packaged in J2EE compliant WAR files
    - Directory structure in the WAR has changed to conform with J2EE
    - Protected resources now in WEB-INF directory

Portlet Application Definition

- In earlier portal versions a portlet application was defined with the portlet.xml document, stored with the portlet application in the portlet application's PAR file
- Portlet Application now descibed in two xml documents
  - portlet.xml continues to be the document to store portlet application and portlet specific information and configuration values
    - eg describes the portlets in an application and their configuration parameters to the portal server
  - web.xml defines the portlet application classes as servlets and includes servlet context initialisation data
    - the "servlet" definitions in a J2EE application server
    - see J2EE Servlet Programming Guide for format
- Compile the Java file and package the contents of these files into a WAR file
  - The deploy WAR file to the Portal Server and install it

Web Deployment Descriptor

- The web application descriptor document defines the specific classes of the portlet application
- The portlet application is defined as a web application
- The web application is identified via a unique UID
- This UID is used to associate the web application in the web application descriptor to the portlet application in the portlet application descriptor
- Each portlet class in the portlet application is defined as a servlet instance
- The servlets are identified by an assigned ID
- The servlet ID is used to cross reference the servlet, as a portlet instance, in the portlet application descriptor

web.xml Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
   2.2//EN"
                    "http://java.sun.com/j2ee/dtds/web-app_2.2.dtd">
<web-app id="com.ibm.samples.helloworld.HelloWorld.1974.1">
 <display-name>Hello World Portlet Application - Portlet Sample #1</display-name>
 <servlet id="Servlet_439329280">
  <servlet-name>HelloWorld</servlet-name>
  <servlet-class>com.ibm.wps.samples.helloworld.HelloWorld</servlet-class>
 </servlet>
 <servlet-mapping id="ServletMapping_439329280">
  <servlet-name>HelloWorld</servlet-name>
  <url-pattern>/HelloWorld/*</url-pattern>
 </servlet-mapping>
</web-app>
```

Portlet Deployment Descriptor

- Portlet applications and the portlets in an application are either concrete or non concrete (Basic definitions)
  - Non concrete portlet applications and portlets are abstract definitions of the portlet application and portlets
    - Defined using portlet-app and portlet tags
    - defines parameters global to all portlet application and portlet instances of this portal app
  - Concrete portlets applications and portlets represent the specific instance of the portlet application and portlets, including initialization context parameters
    - Provides a more initialised form of the portlet app
    - eg Language specific titles, context values etc used for portlet initialisation
    - System administrator changeable values
  - Several concrete portlet applications may derive from a single portlet application
- Event listeners classes are no longer declared in portlet.xml. They are directly implemented at the Portlet class level and the event handling methods are implemented in your portlet class directly

portlet.xml Sample

```
<portlet-app-def>
  <portlet-app uid="com.ibm.samples.helloworld.HelloWorld.1974.1">
    <portlet-app-name>Hello World Portlet Application</portlet-app-name>
    <portlet href="WEB-INF/web.xml#Servlet_439329280" id="Portlet_439329280">
      <portlet-name>HelloWorld</portlet-name>
        <cache>
          <expires>0</expires> <shared>no</shared>
        </cache>
        <allows> <minimized/> </allows>
        <supports>

          <markup name="html"> <view/> </markup>
        </supports>
      </portlet>
    </portlet-app>
    <concrete-portlet-app uid="640682430">
      <portlet-app-name>Hello World Portlet Application</portlet-app-name>
        <context-param>
          <param-name>Portlet Master</param-name>
          <param-value>yourid@yourdomnain.com</param-value>
        </context-param>
        <concrete-portlet href="#Portlet_439329280">
          <portlet-name>HelloWorld</portlet-name>
          <default-locale>en</default-locale>
          <language locale="en_US">
            <title>Hello World - Sample Portlet #1</title>
            <title-short>Hello-Worldd</title-short>
            <description>Hello World - Sample Portlet #1</description>
            <keywords>portlet hello world</keywords>
          </language>
        </concrete-portlet>
    </concrete-portlet-app>
</portlet-app-def>
```

Naming Guidelines for UID's, ID's and HREF Values in the Descriptors

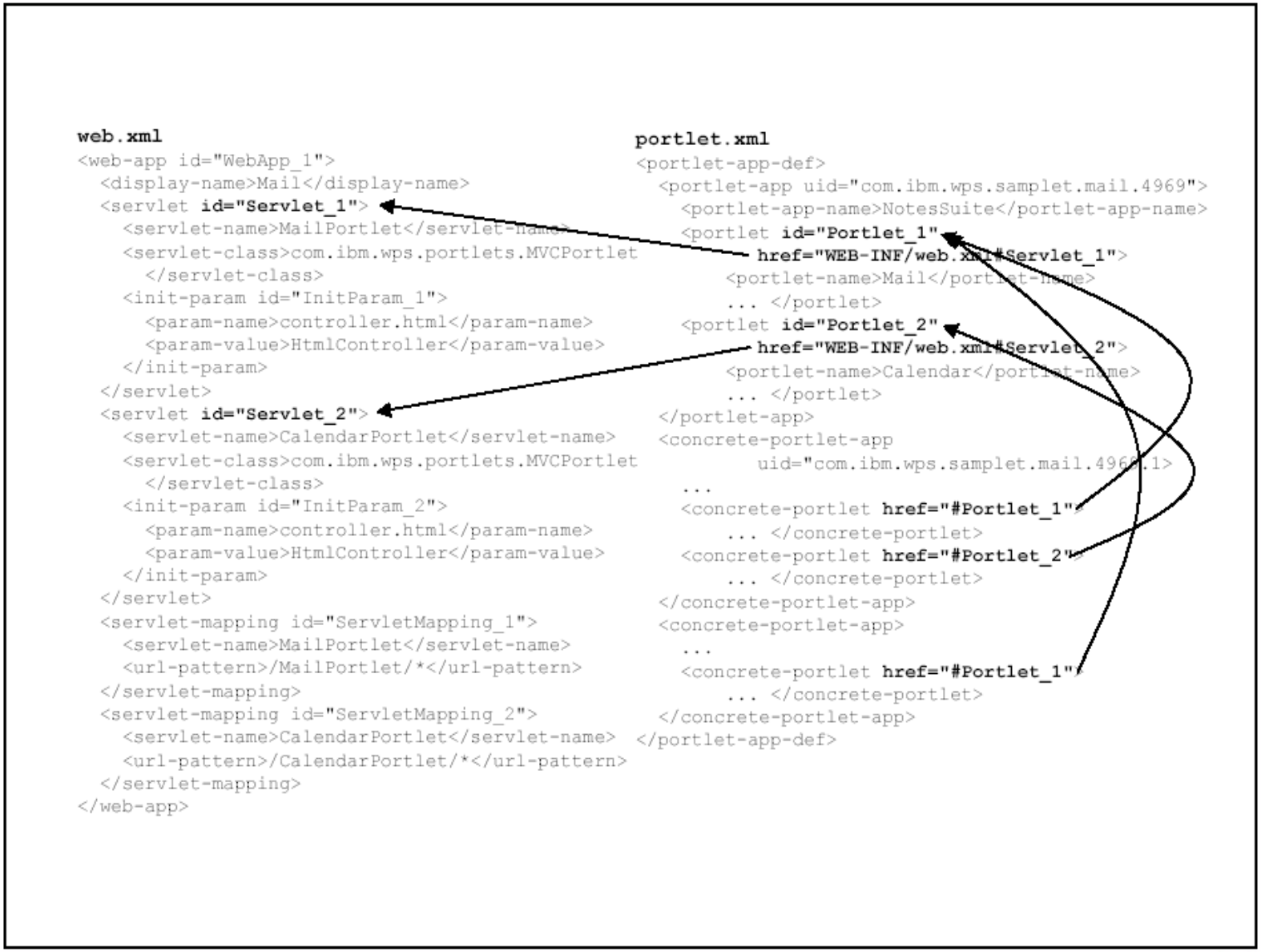
- The UID of a portlet application and concrete portlet application must be unique on all portal systems the portlet applications are intended to be installed and executed. In order to ensure uniqueness the following guidelines are recommended for selecting a UID value:
  - Include the portlet's namespace in the UID, using the same format that is used for Java packages
  - Add a portlet application specific description
  - Add some arbitrary characters to guarantee uniqueness within the namespace, e.g. *com.ibm.wps.samplet.mail.4969*
  - Add postfixes for the corresponding concrete portlet applications, like: *com.ibm.wps.samplet.mail.4969.1*

Naming Guidelines for UID's, ID's and HREF Values in the Descriptors

- Portlet ID values are any valid string, but each ID value for each portlet must be unique within the portlet application.

- Portlet HREF values associate the portlet application descriptor to the servlet definition in the web application descriptor. The href attribute identifies the servlet with the following format: WEB-INF/web.xml#*servlet_id*.

Relating the portlet.xml to the web.xml



The Need For Portlet Migration

- Portlet API (1.0) introduced with WPS 1.2 in 2001
  - Based on Open Source Jetspeed + Extensions
  - Similar to Servlet API + extra functions:
    - Client event handling using standard Java Event handlers
    - Client device info in a standrad format
    - User profile info
    - Portlets can persist settings on a user basis
- The WebSphere Portal Portlet API has evolved to J2EE base and has been updated to include new functions
  - Executes on WebSphere Application Server v4.0.2.
  - Improvements in the portlet inheritance model
    - Portlets inherit from the Servlet Class
  - Portlets can use services offered by the Application Server
  - Java based security
  - Portlet Page Contributions
    - Portlet issues extra style sheets to render portal page
    - Puts in headers (HTTP headers, authentication headers, cookies)

The Need For Portlet Migration

- ►Portlet Dynamic Configurations
  - –Concrete vs non-concrete
  - –Concrete - the Definate Instance
  - –non-Concrete - abstract
  - –Have separate APIs
  - –Administartors update the Concrete Config parameters
- ►Portlet Page Contributions

- IBM is working within the Java Community Process to standardize the portlet container and underlying Portlet API (See JSR 168)
- Portlets developed for earlier released of WebSphere Portal are not binary compatible with WebSphere Portal Version 4.1
- Portlets need to be recompiled and repackaged for WebSphere Portal Version 4.1
  - ►maybe source code changes

Portlet Migration at a Glance

- Portlet Migration is a straightforward effort
- In most cases minimal source code changes are required
- Portlets must be repackaged into a Web Archive (WAR) File
  - IBM provides the PMT tool to assist with migration
  - PMT reads your Portlet application source tree and writes a new source tree converting Java Classes, JSP's, and the Portlet Application Descriptor into the required format for WP 4.1 Portlet Applications
  - The converted portlet application tree can be compiled, packaged into a WAR file and run on a WP 4.1 portal
- IBM also offers a Portlet Migration Guide, which outlines in detail the changes required in most portlet applications

Preparing for Migration

- Install WP 4.1
- Verify the installation and operation of the portal
  - Log In to the portal as a portal administrator.
  - Install a portlet.
  - Add the portlet to a portal page using the customizer.
  - Test the portlet's operation.
- Add the required and optional WP 4.1 Jar files to your JDK/IDE classpath

JAR Filename	Purpose
portlet-api.jar	Contains portlet API class definitions
wpsportlets.jar	Contains AbstractPortlet and MVCPortlet class definitions
j2ee.jar	Contains Servlet engine class definitions
wps.jar	Base class definitions for WebSphere Portal
jlog-2.2.jar	Contains log class definitions

- Compile a sample Portlet
- Create the Web Application Descriptor and Portlet Application Descriptor
  - web.xml, portlet.xml
- Package the compiled Java, web.xml amd portlet.xml into a WAR file
- Deploy WAR file to WP and install it
- Add the portlet to a page and verify it.

WAR Files

- You need to create a directory tree as follows:
  - + Root

    unprotected resources
  - + web-inf
  - + web-inf/portlet.xml
  - + web-inf/web.xml
  - + web-inf/classes/java files
  - + web-inf/lib/jar files
  - + web-inf/html/JSP's
- The WAR must contain the web.xml and portlet.xml files
- All protected resources must be in the WEB-INF directory
- JAR files go in /WEB-INF/lib
- Classes go in /WEB-INF/classes
- PORTLET-INF directory no longer needed

The Portlet Migration Tool (PMT)

- Converts portlet Java classes to WebSphere Portal Version 4.1 API's.
- Converts portlet JSP's to WebSphere Portal Version 4.1 format and adds the the "init" tag to the JSP.
- Converts the portlet application descriptor xml document into the portal's new format and also generates the web application xml document for the portlet application.
- Flags other items in the portlet application that require programmer attention.
- Souce tree
  - + Root
    - + portlet-inf
    - + portlet-inf/portlet.xml
    - + portlet-inf/classes/java files
    - + portlet-inf/html/JSP's
- Target tree
  - + Root
    - + web-inf
    - + web-inf/portlet.xml
    - + web-inf/web.xml
    - + web-inf/classes/java files
    - + web-inf/lib/jar files
    - + web-inf/html/JSP's

WebSphere software

IBM

The Portlet Migration Tool (PMT)

- PMT is a Java Application
  - Need a Java JDK or JRE 1.3.0 installed.
  - Reference pmt.jar and xerces.jar in the classpath.
  - Invoke by:

```
Usage : java PMT [option] -src <source dir> -dst <destination dir> -q

The options that may be specified on the PMT command line:

-c: Convert Java source files to the WebSphere Portal 4.1 API's
-p: Convert Java Server Pages to the WebSphere Portal 4.1 Format
-q: Overwrite files without prompting
-x:  Convert the portlet.xml to new portlet.xml and web.xml
-par: PAR file to WAR file directory conversion
```

But it doesn't do everything (eg Event Handler classes !)

PMT : The Portlet Migration Tool

Sample PMT Command Lines:

Java file conversion

```
java PMT -c -src c:\portlet1\test\PORTLET-INF\classes\com\ibm\wps\portlets
           -dst c:\portlet1\testout\WEB-INF\classes\com\ibm\wps\portlets
```

JSP file conversion

```
java PMT -p -src c:\portlet1\test\PORTLET-INF\html -dst
c:\portlet1\testout\WEB-INF\html
```

XML file conversion

```
java PMT -x -src c:\portlet1\test\PORTLET-INF -dst c:\portlet1\testout\WEB-INF
```

Note: Ensure you have placed a copy of the portlet.dtd in the directory that contains the
portlet.xml file.

Single step PAR to WAR directory conversion

```
java PMT -par -src c:\portlet1\test -dst c:\portlet1\testout
```

PMT : Special Considerations

- Changes all found occurrences of the string “PORTLET-INF” to “WEB-INF”.
- Changes the getSession() method to getPortletSession().
- Completely converts portlets that inherit from MVCPortlet when the –par option is specified.
- Flags Java variables that are declared final but not static.
- Adds the following lines to each JSP:

```
<%@ taglib uri="/WEB-INF/tld/portlet.tld" prefix="portletAPI" %>
<portletAPI:init />
```

The Portlet Migration Tool (PMT)

The PMT tool makes these conversions:

Source Line	Conversion or Message
Portlet implements interface Portlet	Portlet extends AbstractPortlet
Portlet uses org.apache.Jetspeed.portlets.DefaultPortletAction	Portlet uses org.apache.Jetspeed.portlet.DefaultPortletAction
Portlet uses org.apache.Jetspeed.portlets.DefaultPortletMessage	Portlet uses org.apache.Jetspeed.portlet.DefaultPortletMessage
Variable declaration final, but not static	Print error with file and line number
Casts to get HttpSession HttpRequest, or HttpResponse	Remove specific Impl cast since PortletSession, PortletRequest, and PortletResponse are direct descendants of these superclasses.
PortletSession.getUser()	Move to PortletRequest.getUser()
PortletRequest.getSession()	Changed to PortletRequest.getPortletSession()
PortletConfig.getAttribute()	Change to PortletConfig.getInitParamter()
PortletResponse.encodeURI()	Change to PortletResponse.encodeURL()
PortletContext.send()	Changed method parameters, updated toPortletContext.send(String PortletMessage)
ActionEvent.getPortlet()	Deprecated, emit warning message
In JSP:PortletRequest.getSession()	Changed to PortletRequest.getPortletSession()
In JSP:PortletResponse.encodeURI()	Change to PortletResponse.encodeURL()
In JSP: Casts to get HttpSession HttpRequest, or HttpResponse	Remove specific Impl cast since PortletSession, PortletRequest, and PortletResponse are direct descendants of these superclassese.gHttpServletResponse servResponse = ((PortletResponseImpl)response).getServletResponse();To: HttpServletResponse servRes

Other Migration Tools (at GA)

- Requirement to migrate configuration data, databases etc from 2.1 to 4.1
  - Objective is to provide a migration path to allow deployment of WP 4.1 with as little impact to the end usre as possible
  - To preserve as much data as possible
- Migration only supported from 2.1 to 4.1
  - Will need to get to 2.1 first (K-Station usres prior to 1.1 would need to migrate to WP 2.1)
- Documentation will describe various scenarios
  - eg WP 2.1 Enable to WP 4.1 Enable
  - WP 2.1 Extend to WP 4.1 Extend (with and without K-Station and WPS/Bridge)
  - etc
- Documentation will describe reusing existing servers, co-existence, DB2, Secureway upgrades etc

Other Migration Tools (at GA)

- Portal Server Migration Tool:
- Basically, it is a small Java Application that allows the following steps:
  - Uninstall 2.1 portal. Do not delete the database or remove the tables!
  - Start the database migration tool. It will generate a XML file containing the old page- & portlet configuration
  - Install the 4.1 version of the portal (to a different database than the 2.1 portal)
  - Run the XMLConfig tool of WPS to import the generated XML file.
    - Now the page structures should be converted to the new portal, but all portlets are replaced by placeholders.
      - e.g. "Here would be your "Lotus Mail" portlets
  - Convert the portlets to the new PortletAPI interfaces.
  - Deploy the new Portlets to replace the placeholders.
    - Using the update function on the Portlet Administration page
    - retaining the page structure and the portlet configuration (settings/data)

Other Migration Tools (at GA)

- Portal Server Migration Tool:
- What is migrated
  - Portletdata, Portlet Configuration, Extended User Profile, Property Files, Pages, Access Control data
  - Page Groups
    - all shared pages become member of Default Group
    - Mydefault Group for all users personal pages
- What is not migrated
  - Themes, Skins, Portlets, JSPs

Other Migration Tools (at GA)

- K-Station Migration Tool
  - Collaborative Services
    - Migrates Themes, Place data, layout, pages, templates, userdata, directory entries, Sametime info, Portlet types, portlet instances, portal settings, image library
- Bridge Server Migration Tool
- WPS Upgrade Tool
  - Adds Collaboration Components to WPS 4.1 Enable post-installation
- Follow Migration/Upgrade Instructions for
  - Lotus Sametime
  - Site Analyser
  - Domino Extended Search
  - Policy Director
  - Content Manager
  - .....

Portlet Migration: Resources

- Information about WebSphere Portal Version 4.1 is available on the Portal Website: http://www.ibm.com/websphere/portalfamily
- Additional information is available on IBM DeveloperWorks: http://www.developerworks.ibm.com
- Other documentation and tools available from IBM
  - WebSphere Portal Version 4.1 Portlet Development Guide
  - WebSphere Portal Version 4.1 Portlet Migration Guide
  - PMT – The Portlet Migration Tool