What's new in the IBM WebSphere Portal 6.0.1 and 6.1 Programming Model

Stefan Hepper | WebSphere Portal Programming Model Architect
Agenda

- Overview WebSphere Portal programming model
- New APIs and SPIs in WebSphere Portal V6.0.1
- New APIs and SPIs in WebSphere Portal V6.1
- New portlet standards in WebSphere Portal V6.1
- Portlet frameworks
Overview WebSphere Portal Programming Model
Different programming model layers

- **Business users**
  - Goal: develop and customize applications at runtime without programming knowledge
  - Via graphical tools (WebSphere Portal, WebSphere Portlet Factory)

- **Power users**
  - Goal: develop applications without J2EE knowledge
  - Via tools (WebSphere Portlet Factory, Rational Application Developer) and scripting interfaces (Javascript, Dojo, REST services)

- **Application developers**
  - Goal: develop applications with programming knowledge
  - Via Java and web service / REST APIs
  - Tools: Rational Application Developer, WebSphere Portlet Factory

- **ISVs**
  - Goal: customize and enhance the current portal or workplace product with new functionality and services using deep programming knowledge
  - Via Java SPIs (System Programming Interfaces)
Different programming model artifacts

- **Themes and Skins**
  - JSP tag libraries

- **Static HTML**
  - Microformat tags

- **WCM pages**
  - WCM content components

- **Composite applications**
  - UI and business components

- **Portlets**
  - JSR 168 and JSR 286 Portlet API

- **Portlet services**
  - Giving portlets access to WebSphere Portal extensions

- **Eclipse extension points**
  - Customizing existing portal behavior (e.g. Theme extension points)
New APIs and SPIs in WebSphere Portal V6.0.1
Portlet Object Model (SPI)

- Allows you to read-access all data around portlets
  - Portlet deployment descriptor information
  - Different layers of portlet preferences
  - Access to portlet definition, portlet entity, portlet window
    - Unique name, object ID

- Available for themes and skins and standard portlets
Portlet Object Model – Details

- base settings defined by the portlet application developer
- customization settings defined by the portal administrator
- Shared setting defined by the administrator
- personalization settings defined by the portal user
- navigational state representing the current view state of the portlet
- per portlet session state

aggregation of portlet preferences

portlet.xml

portlet definition 1

portlet definition 2

portlet entity 1.1

portlet entity 1.2

portlet entity 2.1

portlet window 1.1.1

portlet window 1.1.2

portlet window 1.2.1

portlet window 2.1.1
Get unique names of portlets on page from within a theme

```java
// lookup service
Context ctx = new InitialContext();
PortletModelHome home = (PortletModelHome) ctx.lookup("portal:service/model/PortletModel");
if (home != null) {
    PortletModel portletModel =
    home.getPortletModelProvider().getPortletModel(aPage, aRequest, aResponse);
}

// use service
for ( // all nodes of the current page ) {
    if (node instanceof LayoutControl) {
        PortletWindow portletWindow =
        portletModel.getPortletWindow((LayoutControl) node);
        PortletDefinition portletDef =
        portletModel.getPortletDefinition(portletWindow);
        String portletUniqueName = portletDef.getObjectID().getUniqueName();
    }
}
```

Lotus software
Resource Addressability Framework (SPI)

- Allows you to address arbitrary content via the portal framework using URIs
  - Allows to keep current navigational state encoded in the portal URL
    - Converts URIs to URLs
  - Allows custom code to define their own content schemes and participate in the "view" resolution

- Address pieces of content (PoCs) via URIs
  - URI guarantees a unique ID for the content
  - The URI that identifies the content is typically not the same as the URL that displays this content
    - There may be many URLs that display the same URI
    - The URL can be used as the URI if there is only one possible content location

- Example
  - Create your own link format to documents in a custom content repository that you want to serve via portal
Resource Addressability Framework – Details

- Resolution out-of-context
  - Client requests a view of a URI
    - Without portal context, e.g. by clicking an e-mail link

- Resolution in-context
  - Client clicks on a link that points to a URI during interaction with the portal
    - E.g. by clicking on a search result entry
And many more ...

- Search menu tag lib (API)
  - Allows adding custom search scopes
- Extension to the credential vault service (API)
  - Support LTPA tokens
- Puma profile now supports external users (SPI)
- ATOM SAX API (SPI)
  - Parser for ATOM feeds in Java
- URL resource service (SPI)
  - For managing URL resource definitions in the WAS config repository
- Login service (SPI)
  - Enabling to write your own login portlet
New APIs and SPIs in WebSphere Portal V6.1
Web 2.0 support

- Allow portlets to connect to external data sources
  - Issue: currently requests are restrict to the same domain because of security reasons
  - Solution: access via provided AJAX proxy

- Client-side aggregation
  - Re-render only changed portlets

- Client-side JavaScript library
  - Convenience JavaScript APIs simplifying portlet development
  - Client-side equivalent to the Java Portlet API
  - Coordinates AJAX calls with the portal
  - Consistent behavior after a full page refresh
  - Navigational state changes
  - Implemented using DOJO
Web 2.0 support

- For more information see Web 2.0 session
Click to Action for everyone (API)

- Enable the Click-to-Action paradigm for standards portlets
  - JSR 168 and JSR 286 portlets

- Based on Web 2.0 semantic tagging technology
  - Thus can be used by any HTML markup (themes and WCM content too)
  - On the browser, scan markup looking for C2A source and target tags, matching sources to targets.
  - generate C2A menus “on-demand”, when the C2A source menu icon is clicked. → more efficient than old C2A

- Integrated with server-side property broker programming model
  - JSR 168 target actions and JSR 286 processing events will be automatically available in C2A (only if they have a single input of type String)
Click to Action for everyone

- Current limitations
  - No complex data types in C2A sources, parameter value must be a string
  - C2A targets must be on the same page as the source
    - No cross-page communication
  - No broadcast functionality
  - No support for automated actions without menu

- Sample Screenshot:
Click to Action for everyone – Example

- **Source tagging:**

  ```html
  <div class="c2a:source someotherclass">
    <span class="c2a:typename" style="display:none">http://www.ibm.com/xmlns/prod/datatype#email822</span>
    <p>some content that is not relevant to C2A
      <b class="c2a:value">johndoe@acme.com</b>
    </p>
    <p class="c2a:header" style="display:none;">
      This is a sample C2A source
    </p>
  </div>
  ```

- **Target tagging:**

  ```html
  <FORM class="c2a:target" action="/myapp/do.something">
    <span class="c2a:typename">http://www.ibm.com/xmlns/prod/datatype#email822</span>
    <p class="c2a:action-label">Show inbox</p>
    Email: <input type="text" class="c2a:paramname"></input>
  </FORM>
  ```
Portal Write Model (SPI)

- Complements the previously introduced Portal Models
  - Enables you to create your own administration portlets, e.g. Eclipse-based
  - In V 6.1 limited to: Content model + navigation model, Layout model, Portlet model

- Available as Java SPIs and REST services

- Based on a workspace concept
  - Apply all your modifications and then do a final commit

- Supported operations
  - Page, label, and ContentURL administration
    - Creating/updating/deleting pages, labels, and ContentURLs
    - Modifying page, label, and ContentURL properties
  - Page layout modifications
    - Adding/deleting/moving portlets (e.g. drag'n'drop) and layout containers
  - Portlet model administration
    - Creating/updating/deleting portlet entities and definitions
  - Property modifications, e.g modifying unique names, meta data
Step-up Authentication (SPI)

- Support/recognize multiple levels of authentication strength ('proof of identity'), e.g.
  - user claim → no proof, cookie
  - username/password → default proof
  - username/password + X.509 Certificate → strong proof

- Progressive disclosure of portal resource
  - Portal resources (pages / portlets) can be flagged to require a specific authentication strength
  - User gets challenged when (s)he's actually accessing the resource

- APIs
  - com.ibm.portal.auth.stepup
    - Allows you to define your own authentication levels
  - com.ibm.portal.portletservice.rememberme.RememberMeCookieService
    - Allows standard portlets to check for the rememberMe cookie
Step-up Authentication – Authentication flow

Anonymous Access

It’s probably Alice

It’s Alice

It’s definitely Alice

Logout

Access Control

Anonymous

Alice

Remember-me Cookie

Username / Password

X.509 Certificate

Identity claim

check balance

transfer money

Image of a scale for Access Control.
Login/logout/session validation Filters

- Allows you to plug into the login/logout/session validation flow of portal
- Use the filter pattern
  - Allows you to provide your own filters
- Provided filters
  - Explicit login by user name and password (com.ibm.portal.auth.ExplicitLoginFilter)
  - Implicit login (e.g. when being already authenticated by WAS) (com.ibm.portal.auth.ImplicitLoginFilter)
  - Explicit logout (com.ibm.portal.auth.ExplicitLogoutFilter)
  - Implicit logout (e.g. after a session timeout) (com.ibm.portal.auth.ImplicitLogoutFilter)
  - Session Timeout (com.ibm.portal.auth.SessionTimeoutFilter)
  - Session Validation (com.ibm.portal.auth.SessionValidationFilter)
Any many more ...

- Property broker SPI
  - Allows you to write your own wiring portlet
- Extend current portlet and portal models to support JSR 286 (SPI)
- Sitemanagement command SPI
  - Allows you to write your own sitemanagement application, e.g. Eclipse-based
- Encoding and decoding of friendly URLs (SPI)
  - Allows you to create friendly URLs and decode friendly URLs
  - Integrates into the resource addressability framework
- Resource Addressability Data Source API (SPI)
  - Allows you to serve your resource addressable data via the default content handler servlet
  - e.g. just define a URI format and add your own handler for serving the URI via the content handler servlet

Lotus software
Any many more ...

- **LocalizedContext API**
  - Allows you to get the preferred locales and titles / descriptions of Localized resources
  - Fixes some bugs that the default JDK ResourceBundles have for calculating the fallback

- **Enhanced PUMA SPI**
  - e.g. ability to do queries for anonymous users with runUnrestricted

- **Multipart Stream Processing SPI**
  - Helper classes to easily process multipart Mime streams and Form data

- **Enhanced portal resource serving SPI**
  - ResourceURLAccessor now allows you to address resources with a relative path and theme specific
  - Use case: package Javascript library (e.g. Dojo) in a specific theme directory and allow the portlet creating links to this lib
    - Removes the need to package the lib with each portlet
New portlet standards in WebSphere Portal V6.1
New Portlet Standards in WebSphere Portal V6.1

- JSR 286 – Java Portlet Specification V2.0
- Web Services for Remote Portlets (WSRP) V2.0

Common goals
- Enable coordination between portlets and allow building composite applications based on portlet components
- Serving resources
- Allow for a better user experience using AJAX patterns

See Portlet Standards session
Portlet frameworks
Portlet frameworks supported by IBM Tooling

- **Java Server Faces (JSF) 1.1**
  - JavaEE standard
  - Rational Application Developer (RAD) V7.0

- **Struts 1.1**
  - IBM Struts Portlet Framework
    - Supports JSR 168 property broker extension
  - Rational Application Developer (RAD) V7.0

- **Portlet Factory**
  - Build portlets based on models and builders instead of fine-grained UI components
Portlet frameworks running on WebSphere Portal

- **JSF V 1.2**
  - Apache JSF Portlet Bridge
  - JSR 301 RI: Standard JSF-Portlet Bridge covering
    - JSR 168 – JSF 1.2
    - JSR 286 – JSF 1.2
    - RI part of Apache JSF Bridge
  - Note: requires Java EE 5.0 which will come with WebSphere Application Server 7.0

- **Apache Struts Portlet Bridge**
  - V 1.x & V 2.0

- **Spring MVC 2.0**

- **Adobe Flex**

- **Apache Wicket**

- **And many more ...**
  - The portlet bridge of the framework just needs to comply to either JSR 168 or 286
Thank you !
Additional Information and Resources

- **WebSphere Portal V6.1 Beta**

- **Websphere Portal Business Solutions Catalog:**
  - http://catalog.lotus.com/wps/portal/portal

- **Webspere Portal Product Information:**

- **WebSphere Portal Information Center Documentation:**

- **JSR 286 information**
  - Reference Implementation: http://portals.apache.org/pluto/

- **WSRP V2.0**
  - http://docs.oasis-open.org/wsrp/v2/wsrp-2.0-spec.html

- **WAS 6.1 WSRP producer**
Additional Information and Resources

- WAS 6.1 WSRP producer
- Resource Addressability Framework
© IBM Corporation 2007. All Rights Reserved.

The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM’s current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

IBM, the IBM logo, Lotus, Lotus Notes, Notes, Domino, Quickr, Sametime, WebSphere, UC², PartnerWorld and Lotusphere are trademarks of International Business Machines Corporation in the United States, other countries, or both. Unyte is a trademark of WebDialogs, Inc., in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Appendix – new APIs in WebSphere Portal V 6.0

- New Themes providing Eclipse extension points
  - Allows separation of WebSphere Portal code from customer code

- Edit_default portlet mode
  - Used by admins to set the default instance values for a portlet on a page.
  - Users can override the default values with their own preferences using edit mode.

- Portlet URL Generation
  - Allow portlets to create URLs to other pages and portlets
  - Allow portlets to get into / out of SOLO state

- Search API
  - Provides a unified API in order to search different IBM backend search products, like Omnifind

- Updated Task processing
  - With Process Server 6.0 the TaskManagerDelegate interface was renamed to HumanTaskManagerDelegate interface
Appendix – new APIs in WebSphere Portal V 6.0

- Updated WCM API
  - Lookup to the WCM service now available via standard JNDI
  - Addition of new component types.
  - Addition of the concept of a library for logically partitioning the content repository into separately managed “mini-repositories”.
  - Addition of referential integrity

- Policy API
  - Policy is a globally scope set of values for a given group of users

- New portlet API services for JSR 168 portlets
  - Inter-portlet communication for JSR 168 portlets
  - Page launching (Dynamic UI Manager)
  - Credential vault service
  - Content access service
Appendix – new SPIs in WebSphere Portal V6.0

- Aggregated Meta Data model SPI
  - Allows for inheritance of MetaData

- Update State handling SPI
  - Use this in portlets only if the URL Generation API is not sufficient
  - Allows to get and set the current locale

- Drag and Drop SPI
  - Allows defining drag sources or drop targets in themes and skins

- New portlet SPI services for JSR 168 portlets
  - Puma SPI
  - Credential Vault SPI
  - Model SPI