Jetspeed -j2o- The Web 2.0 Desktop for Portlets

Jetspeed j2o Desktop is a Web 2.0 solution for portlets. Combining server-side Jetspeed Ajax services with client-side services, j2o is a paradigm shift in the way portlets are viewed and aggregated. The key change here is that Jetspeed 2.0 is a servlet-centric application: every request goes back to the server. The user interfaces experience is driven by HTTP page paradigm.

Jetspeed j2o Desktop is a client-side centric solution. The user interface is controlled on the client-side, where it intuitively belongs. Key features of the portal have been moved to the client side:

1. Page Aggregation
2. Customization

The server-side does what it does best, model and persistence:

1. Stores the model and user information

Client-side Aggregation

Each portlet in its own HTTP request. This request delivers the content for one portlet only. The portlet aggregation process is moved to the client-side.

Client-side Customization

The server stores the portlet definitions and page information. The client-side provides the tools to customize the user's desktop.

Server-side Model

The model is stored on the server-side in standard Jetspeed persistence stores. PSML is served to the client-side, where it is in charge of aggregating the page.

The Desktop

The desktop holds portlets that are displayed to a user. With the first prototype, a desktop is modeled upon a single PSML page.

System Portlets

Each desktop provides by default a set of portlets that are common to all desktops:

1. My Portal (view of the site) Allows you to manage the site's resources that you have privilege to manage: folders, pages, links.

2. Portlet Finder: searches for portlets on the server by keyword, categories. Drag and drop onto desktop supported.
3. My Dashboards – Toolbars are menus of portlets, links and folders, just a different way of grouping.
4. My Links – Organizer and Categorizer for links
5. My Desktop – information about your desktop: change your theme, portlet skins, images, colors, screen saver

Other General Portlets...

1. Weather portlet
2. Calendar
3. Mail
4. ... the usual suspects, see Mac's Dashboard for ideas

Server Side: The Desktop Pipeline

Where do we start?

First, we need to deliver the initial content to the browser. The entry to the Jetspeed j2o Desktop is via the /desktop pipeline, accessed for example as

/jetspeed/desktop

This new pipeline is special in that:

- it does NOT aggregate
- it does not provide Ajax REST payloads
- it only provides the HTML theme to bootstrap the w2o customizer

```
The Desktop Pipelines fires the firing valves:

portalURL -> capability -> security -> localization -> profiler -> desktop
```

The Desktop Valve and Component

```
The Desktop valve is invoked after the Profiler valve.
It requires the page. It simply runs the Desktop component's render method

From the page, we can determine the desktop theme to be displayed from from the
page's skin default (this may be changed in the future)
```

```
<page>
  <defaults
     skin="blue"
...
```

where skin maps into the webapp themes directory

Themes are NOT HTML snippets, they are full HTML pages.
Themes are stored in a directory by name
A theme called "blue" is stored under

```
${WEBAPP-ROOT}/themes/blue
      theme.properties
      theme.vm | .html | .jsp
      /images
      /css
```

Themes can be implemented as JSP, Velocity (vm), or plain HTML
Plain HTML is limited to not using dynamic information.

Typical Theme content:

<HTML>
<HEAD>
$DYNAMIC Javascript Component's content ||
$DYNAMIC Header Component's content

Bootstrap Javascript code to kickoff Javascript engine
</HEAD>
<BODY>
<DIV id='jetspeedDesktop'>

- portlets are dynamically placed here from portlet fragments
- panes are dynamically placed here from layout fragments
- taskbars, toolbars, links (dynamically retrieved)

</DIV>
</BODY>
</HTML>

PSML and the Desktop

PSML contains the instructions on how to draw the desktop.
PSML holds two kinds of fragments:

1. Layout Fragments
2. Portlet Fragments

Layout Fragments map to desktop panes, or in the default case, the single layout fragment container maps to the desktop.
Portlets map to desktop portlet windows.

Eventing

Events are handled by the client-side Javascript.

Some events, such as minimize, maximize, move are sent back to the server.
Events can be queued up and sent back to the server in batch updates.
All events first go through the client-side Event Manager, and then can be propagated to the server side.

Actions and Render URLs

Action and Render URLs must be created by portlets.
The pipeline that generates the portlet will hold a special PortletURL valve that can generate actions that return to the portlet pipeline. Actions should hook into the Javascript eventing system, and then be passed on to the portal. It has been identified that special handling on the server will be required to handle the two phases of the Portlet API, such as the Struts Bridge solution of storing request attributes in the session to span two requests.

Portlet Window Decorators

Window Decorators TBD