Platform-X Page Layout Widgets

This document describes the data structures of Widgets in Platform-X.

# Page Layout Structure

The Platform-X Template course defines the structure in which all data in PX is stored. Homepage widgets and where they are located on a page. The following is a summary of the Template Course structure as it applies to page layout.

**PX\_ROOT**

**PX\_LAYOUT**

**PX\_WIDGETS**

**PX\_PAGES**

**PX\_{page}**

**PX\_{page}\_ZONES**

**PX\_{page}\_ZONE\_1**

**WIDGET\_X**

**WIDGET\_Y**

**WIDGET\_Z**

The folder structure defined above allows PX to layout any number of pages in different ways. The following sections describe the essential Item types that achieve the above structure.

## PX\_LAYOUT

The PX\_LAYOUT item is a standard DLAP Folder item. The parent of PX\_LAYOUT is always PX\_ROOT.

## PX\_WIDGETS

The PX\_WIDGETS item is a standard DLAP Folder item. The parent of PX\_WIDGETS is always PX\_LAYOUT. The purpose of PX\_WIDGETS is to contain the default copy of each widget available in Platform-X. In order to create a new instance of a widget, a copy of one of the widget templates in PX\_WIDGETS is made.

## PX\_PAGES

The PX\_PAGES item is a standard DLAP Folder item. The parent of PX\_PAGES is always PX\_LAYOUT. The PX\_PAGES item acts as a container for all pages that can be configured in Platform-X. These pages include the Home page, and any other pages that allow the user to configure the layout using widgets.

## PX\_{page}

This is a format for how Page items are identified in Platform-X. All Page items exist inside of the PX\_PAGES folder and represent a single logical grouping of widgets and other layout information. The XML for the Page item type is as follows:

## PX\_{page}\_ZONES

Pages are broken down into Zones. Each Zone is a separately configured container for Widgets. As such, the PX\_{page}\_ZONES item is a folder whose parent is the PX\_{page} the Zones inside it belong to.

## PX\_{page}\_ZONE\_{zone}

This is a format string for how to generate item ids for Zone items in a specific page. If you know the name of the page being rendered, you can calculate this item id to get access to the Zone item for a specific zone. Typically {zone} corresponds to an established name for a logical grouping of widgets defined by Platform-X's current design. For example, there are Widgets that exist in the center of the home page so that Zone item's id might be PX\_HOME\_ZONE\_CENTER.

Zone items are essentially folder items with extra metadata that allows them determine which type of widgets they can host. The following is an example XML for a Zone item:

<item id="PX\_HOME\_ZONE\_CENTER" resourceentityid="20723" creationdate="2009-10-30T22:01:58.887Z" modifieddate="2009-11-05T00:20:34.96Z" version="3">

<data >

<!-- Underlying DLAP item type -->

<type >Folder</type>

<!-- Location in template course (in this case the Page it is part of -->

<parent >PX\_HOME\_PAGE</parent>

<!-- Order of the zone -->

<sequence >f</sequence>

<!-- Title of the Zone corresponds to the {zone} part of the

PX\_{page}\_ZONE\_{zone} id -->

<title >CENTER</title>

<abbreviation >CENTER</abbreviation>

<bfw\_type>Zone</bfw\_type>

<!-- Determines which widgets may be placed inside this zone -->

<bfw\_allowed\_widgets>

<widget type="PX\_TOC\_WIDGET" displayName="Table of Contents" />

<widget type="PX\_ASSIGNMENTS\_WIDGET" displayName="Assignments" />

<widget type="PX\_CUSTOM\_WIDGET" displayName="Custom" />

</bfw\_allowed\_widgets>

</data>

</item>

## The Widget Item Type

All Widget items share certain properties that tell the layout engine how and where they should be rendered. The following is an example of item XML that shows these properties and what they do:

<!-- NOTE: The id of a Widget item instance SHOULD NOT be a static value. It should be a GUID similar to any other newly created item in PX -->

<item id="WIDGET\_X" resourceentityid="20723" creationdate="2009-10-30T22:01:58.887Z" modifieddate="2009-11-05T00:20:34.96Z" version="3">

<data >

<!-- Underlying DLAP item type -->

<type>CustomActivity</type>

<!-- Location in template course (in this case the Zone it is part of -->

<parent >PX\_HOME\_ZONE\_CENTER</parent>

<!-- Order of the Widget in the Zone -->

<sequence >f</sequence>

<!-- Title of the Widget is whatever the user chose, or

the default title provided by the Widget's template -->

<title >Title</title>

<abbreviation >TITLE</abbreviation>

<!-- The bfw\_type of all Widgets is Widget -->

<bfw\_type>Widget</bfw\_type>

<!-- The bfw\_subtype tells us what exact type of widget this

instance is and comes from the Widget's template -->

<bfw\_subtype>CustomWidget</bfw\_subtype>

<!-- This element tells the layout engine when it is OK to

render this Widget instance. -->

<bfw\_display\_flags>

<!-- Widget can be displayed to instructors -->

<display>instructor</display>

<!-- Widget can be displayed to students -->

<display>student</display>

<!-- Widget is only shown on the product course -->

<display>productonly</display>

<!-- Widget is only shown on the derrivative course -->

<display>derrivativeonly</display>

</bfw\_display\_flags>

<!-- The element informs the layout engine what controllers and

actions are mapped to the widget interface -->

<bfw\_widget\_callbacks>

<method name="View" controller="CustomWidget" action="View" />

<method name="Edit" controller="CustomWidget" action="Edit" />

<method name=" OnBeforeAdd" controller="CustomWidget"

action=" OnBeforeAdd " />

</bfw\_widget\_callbacks>

<!-- This element gives the item id of the Widget template it was created from -->

<bfw\_widget\_template>PX\_CUSTOMWIDGET</bfw\_widget\_template>

</data>

</item>

# Page Layout Engine

The page layout engine is a set of server and client side components that handler loading, rendering, and general client side behaviors of pages, zones, and widgets.

## Business Layer

The business layer is responsible for loading all necessary data from DLAP and turning it into object consumed by the application's controllers. The following service contract must be defined in the Bfw.PX.Biz.ServiceContracts project. The implementation for this contract should be placed in the Bfw.PX.Biz.Direct.Services project.

### IPageService

**Method:** PageDefinition LoadPageDefinition(string pageName)

**Description:** This method loads the Page item with the id of format PX\_{pageName} along with all of its Zones, and the Widgets in those Zones. Properties of the IBusinessContext must be used to filter the list of Widget items in the Page's Zones based on each Widget's bfw\_display\_flags.

**Method:** void MoveWidget(string pageName, string zoneName, string widgetId, string minSequence, string maxSequence)

**Description:** This method moves the Widget instance so that it is in the specified Zone at the specified position.

**Method:** void SetWidgetDisplay(string widgetId, WidgetDisplayOptions display)

**Description:** This method changes the widget's bfw\_display\_flags values to match the display parameter. This can be used to hide the widget from view in matching cases.

**Method:** void RemoveWidget(string widgetId, string pageName)

**Description:** This method removes the widget from any Zones it is currently in.

**Method:** Widget AddWidget(string pageName, string zoneId, string widgetTemplateId, string minSequence, string maxSequence, string newWidgetId)

**Description:** This method will create a copy of the specified widget and put it in the correct page zone. The newly created Widget will be returned.

**Method:** Widget AddWidget(string pageName, string zoneId, string widgetTemplateId, string minSequence, string maxSequence, IDictionary<string, PropertyValue> properties)

**Description:** This method will create a copy of the specified widget template and then overwrite any properties passed in. The resulting widget instance will be returned.

**Method:** Widget UpdateWidget(string widgetId, IDictionary<string, PropertyValue> properties)

**Description:** This method will update an existing widget by overwriting the properties passed in.

## Application Layer

The application layer is responsible for loading data from the Business Layer and rendering it to a view.

### PageLayoutController

This controller is responsible for loading a PageDefinitionModel and rendering it to a view.

**Action:** ActionResult RenderPage(string pageName)

**Description:** This action method will load the PageDefinition from the IPageService and convert it to a PageDefinitionModel. The resulting model will be rendered by the RenderPage view.