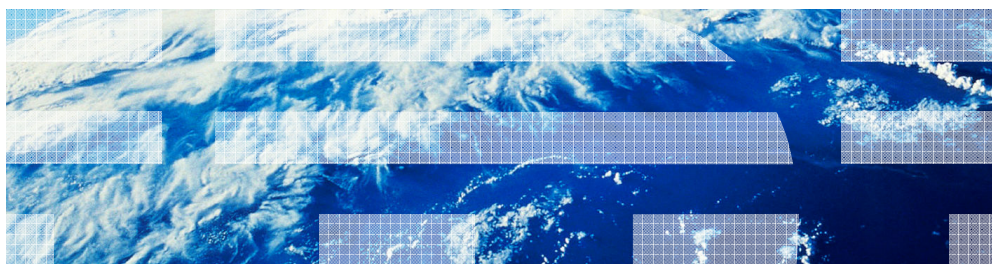




IBM WebSphere Commerce V7 FEP 7

Commerce Composer



WebSphere software

© 2014 IBM Corporation

Agenda

- Solution Overview
 - Business user capabilities
 - Responsive web design
 - Scope
 - Ribbon Ads

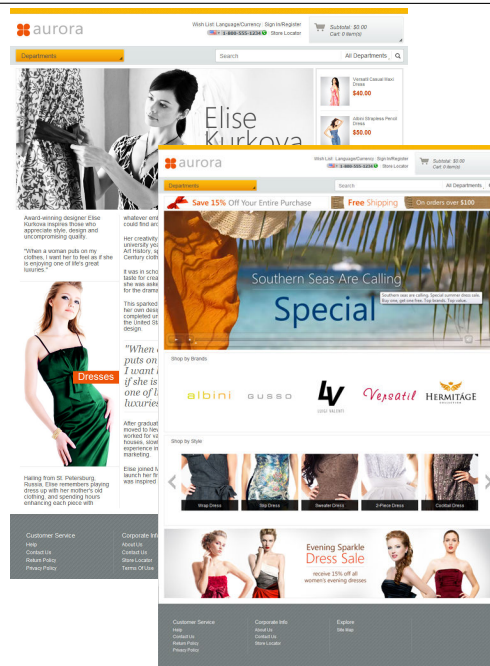
- Demo: Commerce Composer tool

- Commerce Composer Framework Design
 - Terminology
 - Widgets
 - Templates
 - Extended Sites/Content Versioning/Workspaces/Dataload
 - Caching
 - Migration

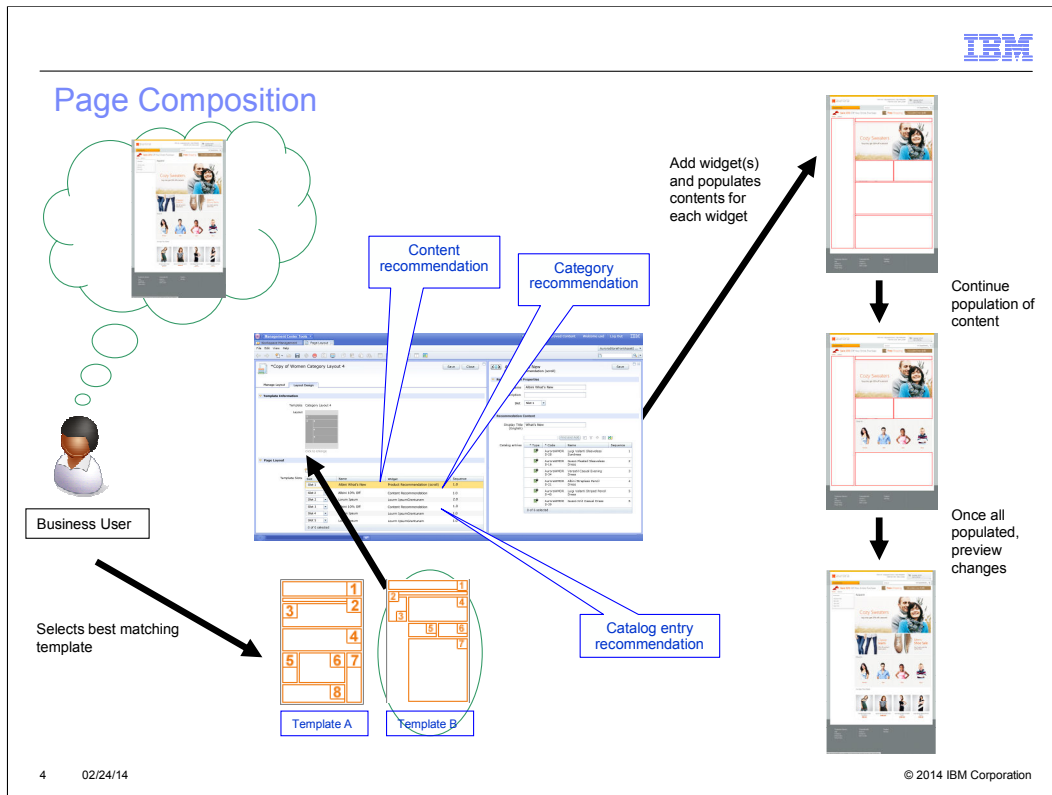
Elevated Business Control

Manage Page Composition & Aggregation

- Build and manage your site pages
- Create new site pages (e.g. landing pages) and optimize URL's and meta-data for pages (SEO)
- Manage customer controls and how content is displayed. Scrolling widgets, hero spots, customer recommendations, etc
- Manage mobile, tablet and web together
- Responsive design for various devices



<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/concepts/cpzpagecomptool.htm>



<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/concepts/cpzpagecompovert.htm>

Commerce Composer Widgets

Widgets

- Content Recommendation
- Category Recommendation
- Catalog Entry Recommendation
- Facet Navigation
- Catalog Entry List
- Links
- E-Marketing Spot
- Content Carousel
- IBM Product Recommendations
- Facebook Activity; Facebook Like
- Text Editor
- Site Map
- Category Navigation
- Site Content List
- Breadcrumb Trail
- Heading
- Search Summary
- Name, Part Number, and Price
- Short Description; Long Description
- Full Image
- Defining Attributes
- Descriptive Attributes

Widgets (Continue ...)

- Merchandising Associations
- Shopper Actions
- Inventory Availability
- Associated Assets
- Discount
- and more ...

Widget Examples

The image shows three examples of Commerce Composer widgets:

- Category Carousel:** Titled "New Summer Arrivals In...", it displays four categories: Blouses, Accessories, Suits, and Skirts, each with a representative image.
- Facet Navigation:** A "Filter by" widget with two sections: "Brand" (listing Abini (8), Guise (8), Lupa Varenk (8), Versall (8), and Heritage Collection (7)) and "Price" (listing "Less than \$100 (31)", "Between \$200 and \$300 (8)", and a price range input field).
- Recommended:** A list of three recommended products: "Get Up and Go! Cafelike Coffee Maker" for \$89.00, "Fit Introive 4 element Electrical Oven" for \$1,199.00, and "Fit Juicer Blender" for \$35.00. A fourth product, "GECC Large Fridge (Element Freezer Style)", is listed for \$1,409.00.

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/concepts/cpzwidgetcont.htm>

IBM Commerce Partners: 3rd Party Widgets

<http://www.ibm.com/commerce-partners>



Amplience Widget
Amplience Widget
[Learn more...](#)



Bazaarvoice Widget
Integrate Bazaarvoice Conversations (Ratings & Reviews, Q&A) into your IBM WCS storefront.
[Learn more...](#)



Fluid Product Customization Widget
Displays product in 3D which enables personalization through attribute (color/texture/monogram) customization
[Learn more...](#)



Genesys Agent
This widget allows displaying a clickable agent icon to allow user to contact an agent.
[Learn more...](#)



BloomSearch widget
This is used to embed BloomReach Organic Search widgets in product and category pages
[Learn more...](#)



CMContentWidget
CoreMedia's Content Widget
[Learn more...](#)



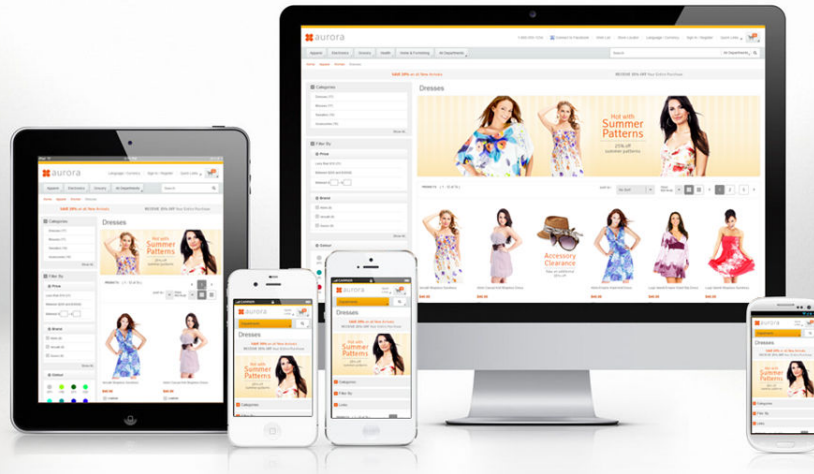
Fluid Shoppable Media Widget
Displays images with hotspots stitched together for autoscroll that engages a user and increases conversion
[Learn more...](#)



OpinionLab Customer Feedback Widget
Displays a persistent (button/tab/link) that shoppers can click to provide real-time feedback about their site experiences.
[Learn more...](#)

<http://www.ibm.com/commerce-partners>

Responsive Web Design – Aurora Starter Store



Advantages of Responsive Web Design

- Overall cost reduction
- Consistent user experience

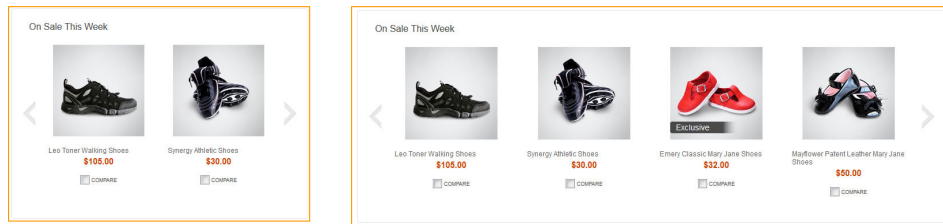
7

© 2014 IBM Corporation

http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.starterstores.doc/concepts/csmaurorarwd_dup.htm

Widget Responsiveness (1 of 2)

- ✓ Widgets respond to screen real estate



http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.aurora-starterstore.doc/refs/rsmaurorasa_rwdwidgets.htm

Widget Responsiveness (2 of 2)

Smartphone (320x480)



Tablet portrait (768x1024)



Tablet landscape (1024x768)



HDTV (1920x1080)



Advantages of eSpot Content:

- ✓ Composes of multiple layers of background images and live text
- ✓ Load appropriately sized background images and position them properly based on screen real estate

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.aurora-starterstore.doc/tasks/tsmaurorarwdmarketingcontent.htm>

Commerce Composer Scope

- Browsing pages
 - Home
 - Department/Category/Sub-category
 - Product/Kit/Bundle/DynamicKit
 - Search results
 - Landing pages/Promotional pages
 - Content pages

- My account and checkout pages
 - NOT in FEP 7

Ribbon Ads - Storefront



GECC Large Front Loading Washing Machine

~~\$1,049.00~~ **\$999.00**

COMPARE



GECC Gas Stove and Oven with 6 Elements

~~\$2,000.00~~ **\$1,849.00**

COMPARE

```
.product_image.RibbonAdDefault {
  position: absolute;
  bottom: 40px;
  left: 0;
  width: 110px;
  height: 20px;
  background-color: #2C2C2C;
  background: -webkit-gradient(linear, left top,
  background: linear-gradient(to right, rgba(68,
  -ms-filter: "progid:DXImageTransform.Microsoft.
  zoom: 1;
  font-size: 14px;
  line-height: 20px;
  font-family: arial, Helvetica, sans-serif;
  color: #FFF;
  padding-left: 10px;
  border-radius: 3px;
  text-align: left;
  vertical-align: baseline;
  overflow: visible;
  white-space: nowrap;
}
.product_image.RibbonAdDefault.Exclusive{
  background-color: #CA4200;
  background: -webkit-gradient(linear, left top,
  background: linear-gradient(to right, rgba(202,
  -ms-filter: "progid:DXImageTransform.Microsoft.
  bottom: 70px;
}
```

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/concepts/cpnribbonad.htm>

Setting Up Ribbon Ads in Management Center

1. Create a Ribbon Ad descriptive attribute

Exclusive
1,2,3

Manage Attribute | References

Comparable

Use as a ribbon ad

Use in merchandising

Use in faceted navigation

Searchable in storefront

Attribute Values Information

Attribute values	Sequence (United States English)	* Type	* Value (United States English)
	1.0		Exclusive

0 of 1 selected

2. Add the descriptive attribute to the product(s)

HAP027_2701

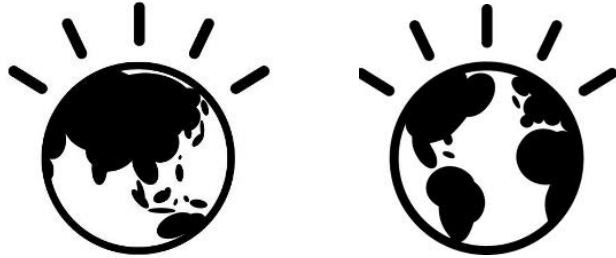
Save and Preview | Save | Close

Manage Product | Search Engine Optimization | **Descriptive Attributes** | Defining Attributes | Merchandising Associations

Descriptive Attributes (Attribute Dictionary)	Sequence	* Type	* Code	* Name (United States English)
	0.0		Finish	Finish
	1.0		Energy Rating	Energy Rating
	1.0		Exclusive	Exclusive
	3.0		Type	Type
	4.0		Dimensions	Dimensions

1 of 5 selected

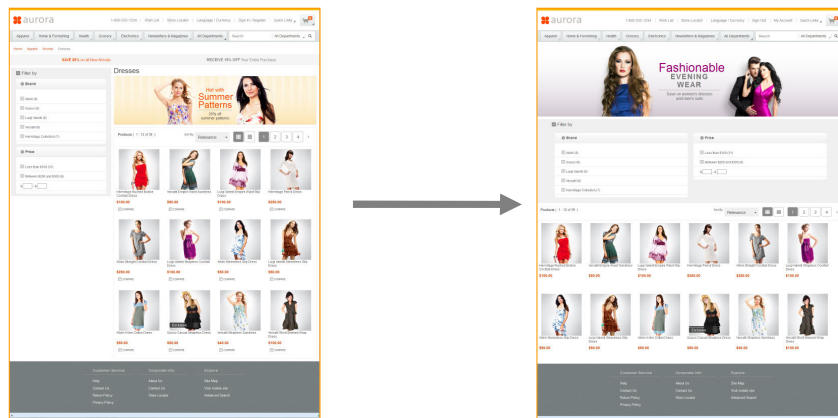
<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/concepts/cpnribbonad.htm>



Commerce Composer Demo

Scenario Walkthrough (Demo of FEP 7)

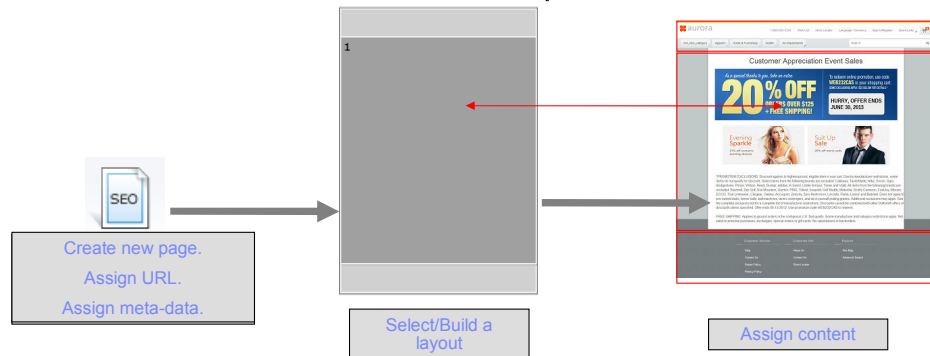
- **Scenario:** Build a new layout for Women's Dresses page
- **Tasks**
 - Business user selects a template to use
 - Business user selects widgets to use for each of the slots



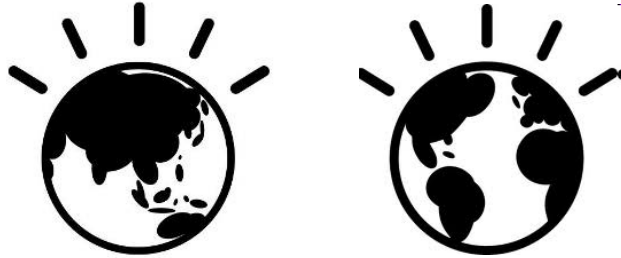
http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/tutorial/tpz_tutindex.htm

Scenario Walkthrough (Demo of FEP 7)

- **Scenario:** Build a new content page for Customer Appreciation Day
- **Tasks**
 - Business user creates a page
 - Business user selects a template to use
 - Business user selects a Content widget to use
 - Business user builds the content using CKEditor (Open Source Web Editor)

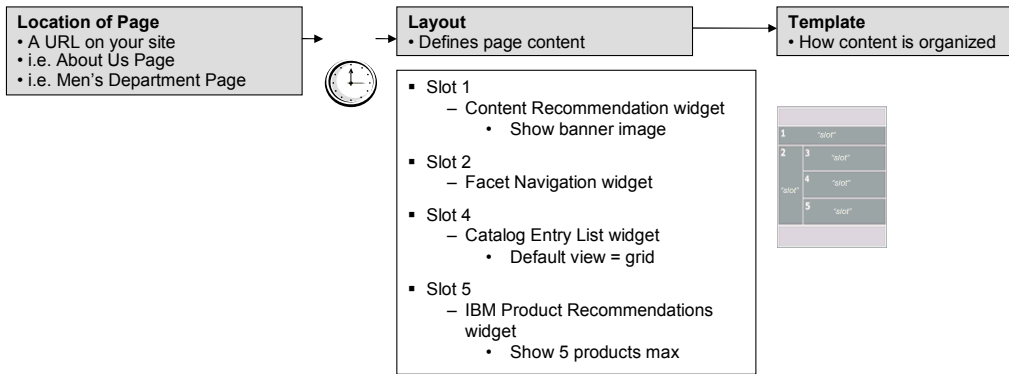


<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.management-center.doc/concepts/ctfckeditor.htm>

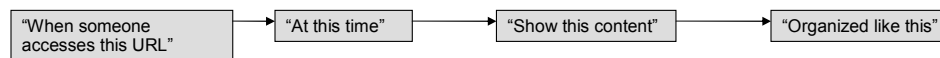


Commerce Composer Framework

Commerce Composer Tool – 3 Concepts



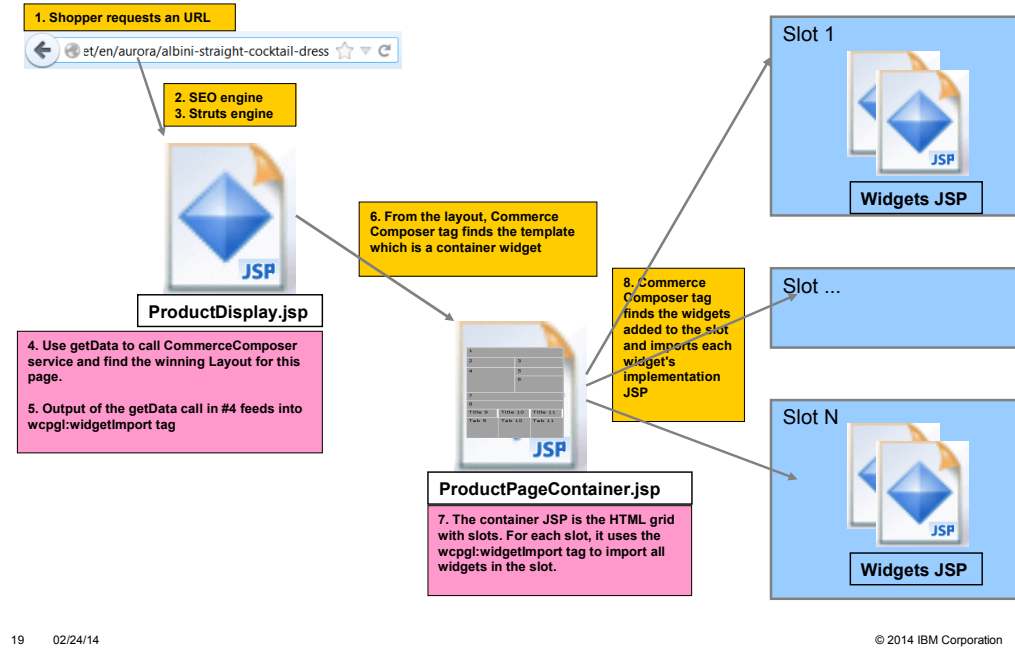
- How to read the above:



More Terminology

- Page – a URL/location in the store.
- Widget – fundamental building blocks of a store page, renders object (s) with pre-defined actions and behaviours.
- Layout – the complete representation of what you see in the store page. It contains meta-data information about what widget (s) to render with its content, how to render them and where to render them in the page.
- Layout assignment – the association between a page in the store and a layout. Its meta-data has scheduling information.
- Template – starting point for creating new layouts, can be considered as an unfinished layout.

Commerce Composer Storefront Architecture



<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/concepts/cpzpagecompflows.htm>

Modify Store JSP Page To Use Commerce Composer Framework

```

1 <!DOCTYPE HTML>
2
3 <wcf:getData type="com.ibm.commerce.pagelayout.facade.datatypes.PageDesignType" var="pageDesign" scope="request" />
4 <wcf:contextData name="storeId" data="{storeId}" />
5 <wcf:contextData name="catalogId" data="{catalogId}" />
6 1. <wcf:param name="deviceClass" value="{deviceClass}" />
7 <wcf:param name="pageGroup" value="Content" />
8 <wcf:param name="objectIdentifier" value="{plPageId}" />
9 <c:forEach var="aParam" items="{wcfParamValues}">
10 <wcf:param name="{aParam.key}" value="{aParam.value}" />
11 </c:forEach>
12 </wcf:getData>
13
14 <wcf:getData>
15
16 <set var="PAGE_DESIGN_DETAILS_VAR" value="pageDesign" scope="request" />
17
18 <html lang="{shortLocale}" xml:lang="{shortLocale}">
19 <head>
20 <link rel="stylesheet" href="{jspStoreImgDir}{env_vfileStylesheet}" type="text/css" />
21 <% include file="../Common/CommonJSToInclude.jspf" %>
22
23 <wcpgl:jsinclude varPageDesignDetails="pageDesign" /> 3.
24 </head>
25
26 <body>
27 <div id="page">
28 <div id="headerWrapper">
29
30 </div>
31
32 <div id="contentWrapper">
33 <div id="content" role="main">
34 <set var="rootWidget" value="{pageDesign.widget}" />
35 2. <wcpgl:widgetImport uniqueId="{rootWidget.widgetDefinitionIdentifier.uniqueId}" debug=false />
36 </div>
37 </div>
38
39 <div id="footerWrapper">
40 </div>
41 </div>
42 <c:if test = "{!empty plPageId}">
43 <wcpgl:pageLayoutCache pageLayoutId="{pageDesign.layoutId}" pageId="{plPageId}" />
44 </c:if>
45 </html>
20 03/04/14

```

1. Call the new PageDesign service to resolve the layout to use for this page.

Input to PageDesign service:
pageGroup – can be Category, CatalogEntry, Content
objectIdentifier – can be categoryId, productId, pageId respectively
deviceClass – can be Web, Mobile

2. Use the widgetImport tag to import the container widget

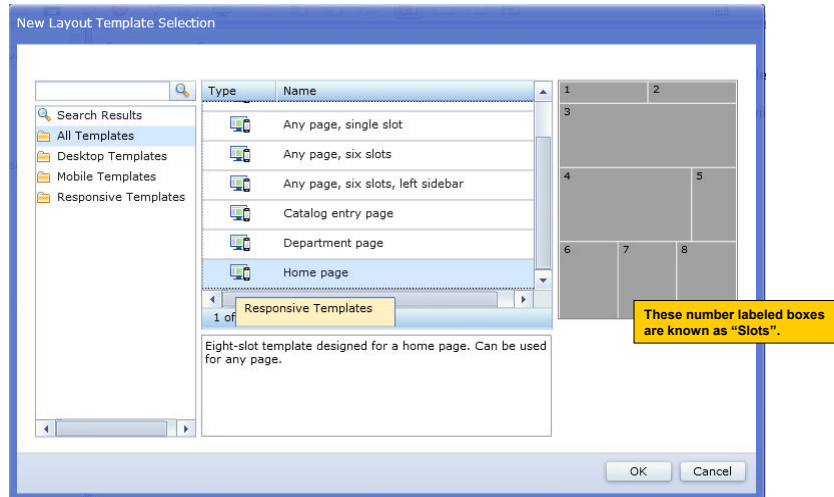
3. Use the jsinclude tag to generate all the JavaScript include tags

© 2014 IBM Corporation

http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/tutorial/tpz_createsitewidget_modifyjsp.htm

Templates

- The first step of creating a Layout is for the business user to select a Template.



<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/concepts/cpztemplateframework.htm>

Steps for Creating a Template – Storefront Part

1. Create a container widget (steps are same as a regular widget)

➤ Store JSP that mainly has the structure of the page

```

<%@include file="../Common/EnvironmentSetup.jspf" %>
<%@ taglib uri="http://commerce.ibm.com/pagelayout" prefix="wcppl" %>

<div id="headerWrapper">
  <%out.flush();%>
  <c:import url = "${env_jspStoreDir}/Widgets/Header/Header.jsp" />
  <%out.flush();%>
</div>

<div id="contentWrapper">
  <div id="content" role="main">
    <div class="row">
      <div class="col6 mcol12" data-slot-id="1"><wcppl:widgetImport slotId="1"/></div>
      <div class="col6 mcol12" data-slot-id="2"><wcppl:widgetImport slotId="2"/></div>
    </div>
    <div class="row">
      <div class="col12 mcol12" data-slot-id="3"><wcppl:widgetImport slotId="3"/></div>
    </div>
    <div class="row">
      <div class="col8 mcol12" data-slot-id="4"><wcppl:widgetImport slotId="4"/></div>
      <div class="col4 mcol12" data-slot-id="5"><wcppl:widgetImport slotId="5"/></div>
    </div>
    <div class="row">
      <div class="col4 mcol12 dcol4 left" data-slot-id="6"><wcppl:widgetImport slotId="6"/></div>
      <div class="col4 mcol12 dcol4" data-slot-id="7"><wcppl:widgetImport slotId="7"/></div>
      <div class="col4 mcol12 dcol4" data-slot-id="8"><wcppl:widgetImport slotId="8"/></div>
    </div>
  </div>
</div>

<div id="footerWrapper">
  <%out.flush();%>
  <c:import url = "${env_jspStoreDir}/Widgets/Footer/Footer.jsp" />
  <%out.flush();%>
</div>

```

Use the new widgetImport tag to import all the widgets setup by the business user in each slot

•FEP5 include a widget: <c:import url = "\${env_jspStoreDir}/Widgets/ESpot/ProdRecommendation.jsp" />
 •FEP7 including a widget: <wcppl:widgetImport slotId="4"/>

Steps for Creating a Template – Storefront Part (cont'd)

- Register container widget in Commerce Composer framework in PLWIDGETDEF table

- Subscribe the store to use the container widget in PLSTOREWIDGET

- Container widgets do not require Management Center object definition nor properties view definition

Register Template In Commerce Composer Framework

1. Register the template with an entry in PAGELAYOUT table and set the isTemplate to true.
1. The template must have a root widget that is a container widget. Register the widgets that the template has in PLWIDGET and PLWIDGETREL.
1. Describe the container widget for Management Center to display the template thumbnail. Each slot of the container widget needs to be described with exact positions and unitless dimensions.

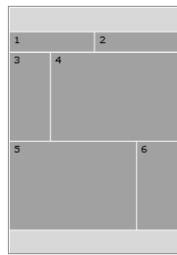
SLOTID	PROPERTIES
'Header'	'xLocation:0;yLocation:0;width:166;height:24;state:disabled'
'1'	'xLocation:0;yLocation:24;width:83;height:20'
'2'	'xLocation:83;yLocation:24;width:83;height:20'
'3'	'xLocation:40;yLocation:44;width:126;height:54'
'4'	'xLocation:0;yLocation:44;width:40;height:174'
'5'	'xLocation:40;yLocation:98;width:126;height:120'
'Footer'	'xLocation:0;yLocation:218;width:166;height:23;state:disabled'

Aurora Template List

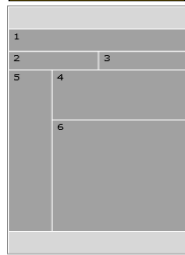
1. Any page, Single slot



2. Any page, six slots



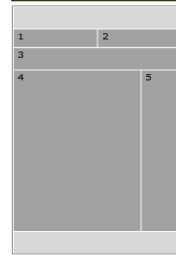
3. Any page, six slots, left sidebar



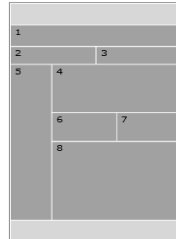
4. Any page, eighth slots, tabs



5. Any page, five slots, right sidebar



6. Home page



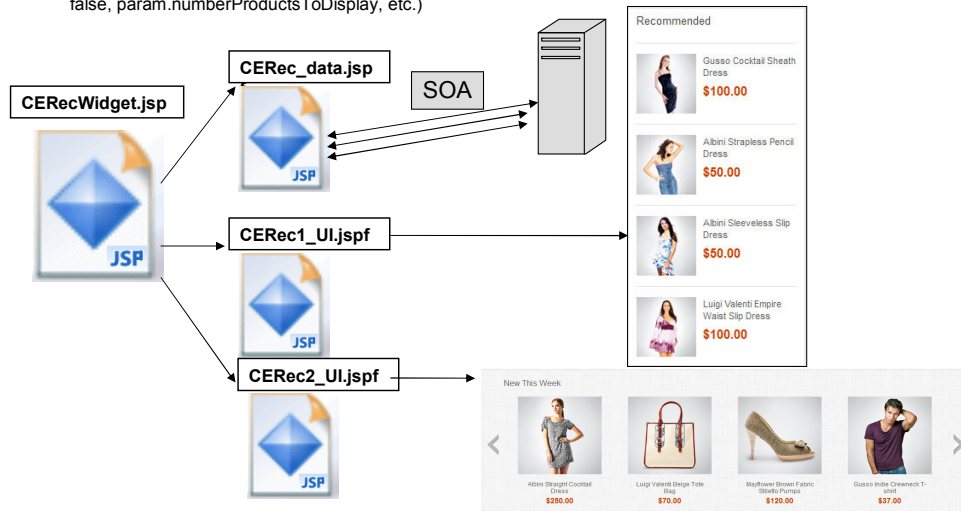
7. Department page



8. Catalog entry page

Steps for Creating a Widget – Storefront Part

- Write a store JSP
 - Used to render the widget in storefront
 - Any rendering effects the widget may have, expose them as JSP import parameters (i.e., param.orientation - true or false, param.numberProductsToDisplay, etc.)



<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/concepts/cpzwidgetframework.htm>

Steps for Creating a Widget – Storefront Part

- Register the Widget with the Commerce Composer framework in PLWIDGETDEF
 - WidgetType – values of 1 for widget or 2 for container widget
 - Vendor – who provided the widget
 - JSPPath – path to the JSP relative to Stores.war
 - DefinitionXML – define the properties/parameters that the widget has. Another important property is **_pgl:javaScriptInclude**, it tells the framework to include these JavaScript files that the widget depends on.

- Stores need to subscribe to the widget in order to use it on layouts. The subscription is done in PLSTOREWIDGET table.

A Closer Look At Widget DefinitionXML

```

<Definition>
  <widget-property name="viewName">
    <value>miniGrid</value>
  </widget-property>

  <widget-property name="enseName" required="false" datatype="java.lang.String"/>
  <widget-property name="numberProductsToDisplay" required="false" datatype="java.lang.Integer"/>
  <widget-property name="showFeed" required="false" datatype="java.lang.Boolean"/>

  <widget-property name="_pgl:javascriptInclude">
    <value>${jsAssetsDir}javascript/Widgets/ShoppingList/ShoppingList.js</value>
    <value>${jsAssetsDir}javascript/Widgets/QuickInfo/QuickInfo.js</value>
    <value>${jsAssetsDir}javascript/Common/ShoppingActionsServicesDeclaration.js</value>
    <value>${jsAssetsDir}javascript/Widgets/swipe.js</value>
    <value>${staticAssetContextRoot}/Widgets/Common/javascript/Recommendation.js</value>
    <value>${staticAssetContextRoot}/Widgets/com.ibm.commerce.store.widgets.CatalogEntryRecommendation/javascript/CatalogEntryRecommendation.js</value>
  </widget-property>
</Definition>

```

1. You can define properties with hard coded values

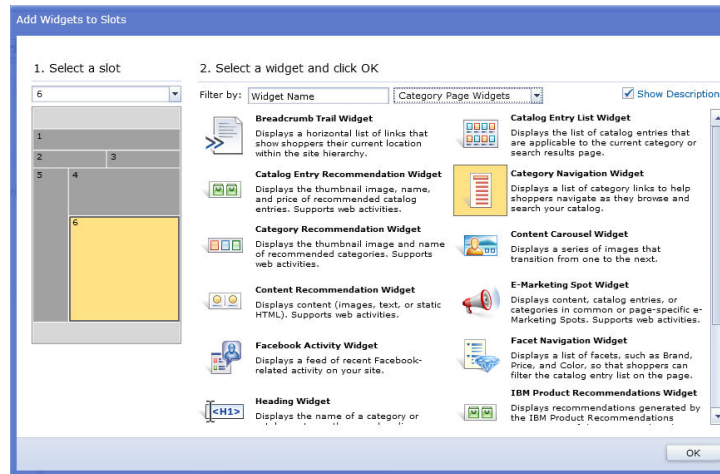
2. You can define properties that are to be set by the business user. It may or it may not have a value.

3. Define the JavaScript files that the widget depends on. The framework will make sure to include this JS file only once in the page.

Note: you can use any variable that is available in the JSP page.

Management Center View For Selecting Widgets

- When creating layouts, business users see the list of widgets from the widget library dialog and assign it to a slot in the layout.
- The widget library list comes from the widgets that the current store has access to.
- Widget library is categorized by widget groups that the widgets are allowed to be used in (ex: Category page, Search results page, Product page, etc)
- Type ahead feature to find widgets easily (within the context of the selected category)



Steps for Creating a Widget – Management Center Part

Layout slots and widgets

* Slot	Sequence	Widget Name	Widget Type
1	0.0	Catalog Entry Recommendation Widget	Catalog Entry Recommendation Widget

Widget Properties

*Widget name ⓘ
 Catalog Entry Recommendation Widget

Widget orientation ⓘ
 Horizontal
 Vertical

Show subscribe link

Widget Content

Select a method for populating this widget
 Specify a list of catalog entries
 Use a web activity to recommend catalog entries

Display Title (United States English)

Catalog entries
 Find and Add

* Type	* Code	Name	Sequence
0 of 0 selected			

1 of 1 selected

Used to save properties that will be passed to the storefront.

1. Define a child object (object definition)

- Declare the properties view def file
- Declare the icon file for the widget
- Define the properties to send to server
- Define create and update services

2. Define a properties view (properties definition)

- Render the entire widget properties view to gather information needed by the widget from business user

Widget Properties View Definition – Management Center Part

Widget Properties

*Widget name
 1

Widget orientation
 Vertical 2
 Horizontal

Widget Content 3

This widget automatically retrieves and displays the facets that shoppers can use to filter the catalog entries listed on the current page.

```

<?xml version="1.0" encoding="UTF-8"?>
<Definitions>
  <GridObjectProperties definitionName="plmWidgetProperties_FacetNavigationWidget">
    <PropertyPane>
      <PropertyGroup name="widgetProperties" collapsable="false" groupTitle="{plmPageLayoutResources.widgetPropertiesPrompt}">
1
        <PropertyInputText name="{plmPageLayoutResources.widgetNamePrompt}" propertyName="widgetName"
          promptText="{plmPageLayoutResources.widgetNamePrompt}" />
2
        <PropertyRadioGroup propertyName="{plmPageLayoutResources.widgetOrientationPrompt}" />
      </PropertyGroup>
3
      <PropertyGroup name="contentProperties" collapsable="false" groupTitle="{plmPageLayoutResources.widgetContentPrompt}">
        <PropertyStaticText text="{plmPageLayoutResources.facetNavigationWidgetContent}" />
      </PropertyGroup>
    </PropertyPane>
  </GridObjectProperties>
</Definitions>
  
```

Widget Object Definition – Management Center Part

```

<?xml version="1.0" encoding="UTF-8"?>
<Definitions>
  <WidgetObjectDefinition package="plm"
    definitionName="plmLayoutWidget_FacetNavigationWidget"
    baseDefinitionName="plmBaseLayoutWidget"
    parentDefinitionName="plmBasePageLayoutPrimaryObjectDefinition"
    objectType="FacetNavigationWidget"
    gridPropertiesDefinitionName="plmWidgetProperties_FacetNavigationWidget"
    iconPath="/images/pagelayouts/widgetIcons/facet_nav.png"
    widgetDisplayGroups="Category,Search"
    widgetRestrictionGroups="Category,Search">
    <PropertyDefinition propertyName="xWidgetProp_widgetOrientation">
      <PropertyValue displayName="{plmPageLayoutResources.facetNavigationWidgetWidgetOrientation_Vertical}" value="vertical" />
      <PropertyValue displayName="{plmPageLayoutResources.facetNavigationWidgetWidgetOrientation_Horizontal}" value="horizontal" />
    </PropertyDefinition>
    <CreateService baseDefinitionName="plmBaseCreateLayoutWidget" />
    <UpdateService baseDefinitionName="plmBaseUpdateLayoutWidget" />
    <Xml name="template">
      <xWidgetProp_widgetOrientation>vertical</xWidgetProp_widgetOrientation>
      <sequence>0</sequence>
    </Xml>
  </WidgetObjectDefinition>
</Definitions>

```

Declare the properties view definition name and path to the icon file.

Specify which widget group to belong into (optional)

Widget Property Manager

- Greater fine grain control for widget properties in read (storefront) and write (Management Center) scenarios as they are persisted to PLWIDGETNVP table.
- A widget can declare its own logic for validation, persistence and retrieval of widget properties. If a widget does not declare its own Widget Property Manager class then the default logic would apply which is no validation and save/read all properties from PLWIDGETNVP table.
- Save
 - All properties from Management Center are passed to the widget specific class
 - Process as required, pass back a new set of properties
 - The new set of properties remaining will be saved in PLWIDGETNVP
- Read
 - All properties are passed to the widget specific class
 - Process as required, and pass back the set that will be sent to the widget JSP in store

Java Emitter Template (JET) package Transformation

- Generates all the necessary source code and directory structure for creating a new widget
 - CSV files for registering widget and subscribing store to widget
 - Management Center files to display widget properties in Commerce Composer Tool
 - Store JSPs for display of widget contents

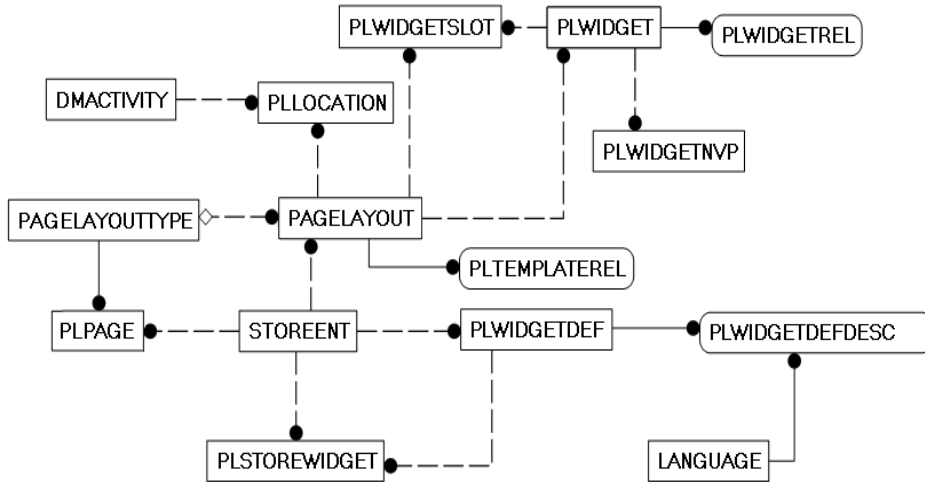
- Input file specifies information about your widget
 - Ex: vendor name, widget name, widget description, widget properties, etc

- Plugin is already part of the WebSphere Commerce Developer environment and can be executed from your workspace

- See Tutorial:
<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/tasks/tpzwidgetcreatesrccd.htm>

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/tasks/tpzwidgetcreatesrccd.htm>

Commerce Composer Data Model



<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.data.doc/ref/s/rdmcommercecomposerdatamodel.htm>

Commerce Composer Support for Other Components

- ✔ Extended Sites – different stores can define a different layout
- ✘ Content versioning
- ✔ Workspaces – complete workflow with approvals
- ✔ Dataload
 - New widgets or update widgets
 - Widget subscription for a store
 - Templates
 - Default layouts – excludes widget content
 - Layouts – excludes widget content

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.data.doc/refs/rmlbestpraccommercecomposer.htm>

Layout JSP Caching And Invalidation

- Full page servlet caching is now affected by business user changes to Commerce Composer layouts
- For cache invalidation to work properly as layouts are modified or as new layouts are applied to pages, you must add 'PageDesignMetaDataGenerator' to the servlet cache entry in the cachespec.xml file
- PageDesignMetaDataGenerator does the following:
 - Determine if the layout can be cached
 - Add 'emsName:name' and 'pageLayoutId:layoutId' dependency ids to the cache entry

```
<cache-id>
  <component id="" type="pathinfo">
    <required>true</required>
    <value>/ProductDisplay</value>
  </component>
  <component id="storeId" type="parameter">
    <required>true</required>
  </component>
  ....
  <component id="productId" type="parameter">
    <required>true</required>
  </component>
  <component id="categoryId" type="parameter">
    <required>false</required>
  </component>
  <metadatagenerator>com.ibm.commerce.pagelayout.cache.PageDesignMetaDataGenerator</metadatagenerator>
</cache-id>
```

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/concepts/cpzcccaching.htm>

Widget JSP Caching And Invalidation

- Some widgets maybe better cached separately as JSP fragments, for example: Header widget, Footer widget, etc. Nothing new.
- Some widgets are very dynamic and cannot be cached, for example: Inventory widget, etc. Declare these JSPs as do-not-cache in cachespec.xml same as before.
- Some widgets can be sometimes dynamic and sometimes static...dynamically determine if they are static and can be cached by parent, or they are dynamic and cannot be cached or they can be cached separately as fragments. For example, espot or recommendations widgets.

```
<metadatagenerator>  
  com.ibm.commerce.marketing.cache.EMarketingSpotMetaDataGenerator  
</metadatagenerator>
```
- Some widgets are static and can be consumed by the parent servlet caching. However, since the widget can be placed on any page, then the invalidation need to be setup dynamically.
 - <dependency-id>ignoreDoNotConsume</dependency-id>
 - <wcpgl:pageLayoutWidgetCache/>

<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/concepts/cpzcccaching.htm>

How To Adopt Commerce Composer?

1) Use FEP 7 Aurora as a starting point

- New customers
- Existing customers looking for major site redesign and willing to start over (<http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.aurora-starterstore.doc/tasks/tsmmigrate.htm>)

1) Use FEP 7 Aurora templates and widgets

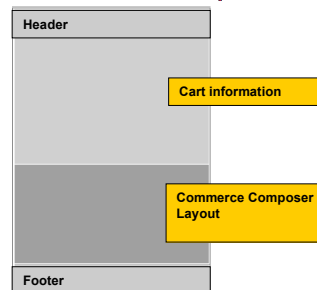
- Add a new store function called 'UseCommerceComposer'
- (If coming from PageLayout tool) Remove store function called 'UsePageLayout'
- Subscribe to site level widgets
- Dataload the Aurora templates or create new templates
- Modify main pages from the browsing pages to call Commerce Composer tags
- Dataload default layouts
- Keep custom store as is specially My Account and Checkout

2) Convert custom store into templates and widgets

- Add a new store function called 'UseCommerceComposer'
- (If coming from PageLayout tool) Remove store function called 'UsePageLayout'
- Follow info center instructions on how to create templates and widgets
- There is a tutorial that converts shopping cart page to Commerce Composer (http://pic.dhe.ibm.com/infocenter/wchelp/v7r0m0/topic/com.ibm.commerce.pagecomposerframework.doc/tutorial/tpz_createsitewidget_introduction.htm)

Best Practices

- Want manage the layout of another store page in Commerce Composer?
 - Any page can be managed in Commerce Composer
 - Represent the store page as a Content page
 - Mark the Content page as 'url not configurable'
 - Is not necessary to have the entire page open for business users
 - Have hardcoded widget or code in the main JSP page, and then just use the `wcpgl:widgetImport` tag in the preferred area



Best Practices (cont'd)

- Default layouts?
 - These layouts are not managed nor visible in Management Center
 - Every page group has a default layout which is defined by IT via dataload
 - Out of the box page groups: Content, CatalogGroup, Product, SKU, Bundle, Kit, DynamicKit, SearchResult
 - In Aurora, default for CatalogGroup page is the layout that you see for dresses and all leaf categories. Therefore, you will see that departments and category defines new layouts and do assignments to the select categories
 - In Aurora, default for Content page is the layout that you see for Help page, ContactUs page, etc...therefore for Home and Sitemap pages we define new layouts and do assignment for that page

Thank You!