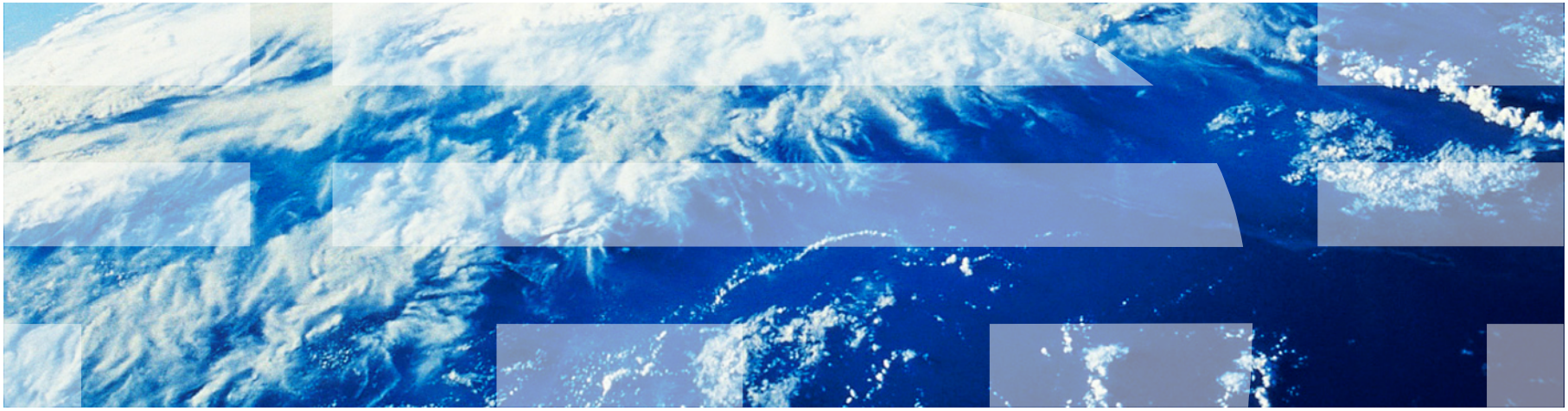
IBM WebSphere Commerce V7 FEP7

Responsive Web Design & Mobile

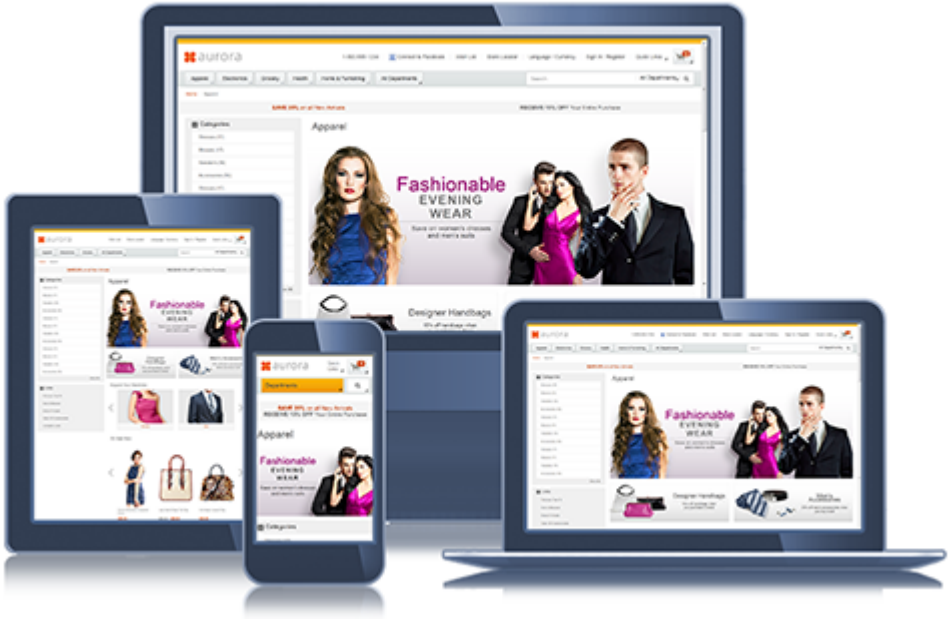


Agenda

- What is RWD?
- Why RWD?
- Goals of WC RWD
- What's new/changed in FEP7?
- RWD Features
- Responsive grid system
- Supported devices
- Store assets
- Customization
  - Creating responsive layout templates
  - Creating responsive widgets
  - Creating repsonsive marketing contents

What is RWD?

- RWD is a web design approach aimed at crafting pages that are optimized for a wide range of devices. It advocates the use of primarily fluid layouts and media queries to optimize a site for different devices, instead of designing a separate site for each device.



Why RWD?

- Better user experience
- Saves time and money
- Future-proof

Goals of WC RWD

- Use industry standards like fluid layouts and media queries to make the Aurora store responsive
- Provide rich, retail-centric responsive page templates that are accessible, localized and touch-friendly
- Give business users complete control of page content and layout
- Provide workflow to design, assemble and test responsive pages

What's new/changed in FEP7?

- Most browsing pages in the Aurora store are now fully responsive/Commerce Composer-enabled:
  - Home page
  - All department pages
  - All categories pages
  - All subcategories pages
  - Women's hats and glasses search landing page
  - Search results page
  - All product/package/bundle/kit details pages
- All OOTB layout templates and widgets (new in FEP7) are fully responsive
- Some pages are now populated with responsive marketing contents:
  - Home page
  - Apparel department page
  - Women category page
  - Dresses subcategory page
  - Women's hats and glasses search landing page

What's new/changed in FEP7? (cont'd)

- The rest of the Aurora store pages will continue to be device-specific:
  - AuroraMobile.sar is now part of Aurora.sar
  - Device detection/forwarding is enabled OOTB
  - Aurora desktop pages are made fluid to support different tablet sizes
  - Key parts of the Aurora desktop/tablet and mobile pages (e.g. header and footer) have been updated to match the L&F of the responsive pages
- CMC store preview tool can now preview store pages in different viewport sizes to simulate how they look on different smartphone/tablet sizes

RWD features

- Supports all viewport widths >= 320px (in logical pixel*)
- Key viewport width ranges:
  - RWD-A: 320-600px
  - RWD-B: 601-1280px
  - RWD-C: 1281px+
- Responsive header and footer:
  - RWD-B/C: more quick links and department dropdowns will be shown as more horizontal space becomes available
  - RWD-A: the department dropdowns will collapse into a single departments dropdown with an expandable section for each department
  - RWD-A: the search box will collapse into a search button which shows the the search box when clicked
  - RWD-A: the footer sections will collapse into expandable sections
- Fluid layouts with column stacking in RWD-A:
  - Based on a 12-column responsive grid system
  - Slots will resize horizontally to fit the viewport
  - RWD-A: slots will stack to show widgets in a single column
    - e.g. sidebar widgets will collapsed into expandable sections and stacked with other widgets in a single column

*Logical pixel vs physical pixel: http://ariya.ofilabs.com/2011/08/mobile-web-logical-pixel-vs-physical-pixel.html

RWD features (cont'd)

- Responsive widgets:
  - Widgets will resize horizontally to fit the slots
  - The recommendation widgets will adjust the number of items per page based on slot width with touch support
  - The catalog entry list widget will adjust the number of items per row based on slot width
- Responsive marketing contents:
  - Marketing contents will resize to fit the slots
  - Some marketing contents are fully responsive and will adjust their compositions based on viewport width

Responsive grid system

- The Aurora store comes with a 12-column, fluid grid system with responsive features used by the OOTB layout templates
- Basic example of a row with 2 columns spanning 4 and 8 spaces:
  -

- Columns can respond to RWD-A and RWD-C by using one of the acol* (for RWD-A) and/or ccol* (for RWD-C) classes. Specifically, the following change will enable column stacking in RWD-A:
  -

Supported devices

- Browser Support:
  - IE10
  - Firefox (17esr+)
  - Chrome (same as FEP6)
  - Safari (same as FEP6)
- With touch support:
  - iOS5+ (smartphones and tablets)
  - Android (v2.3, 4.0+) on smartphones
  - Android (v4.0+) on tablets
  - WP8
  - IE10 on Win8/RT w/touchscreen
- With non-functional degradations:
  - IE8 - RWD-B only, no rounded corners, no transition animation
  - IE9 - no transition animation

Store assets

- Refer to the Commerce Composer session for more details
- Stores.war/<storedir>/Container/*.jsp
  - Container JSP
  - Each referenced by a container definition in the PLWIDGETDEF table
  - Divides a page into slots using our responsive grid system and imports the widgets assigned to each slot using the <wcpgl:widgetImport> tag
  - Does NOT include HTML boilerplate (e.g. <html>, <head> and <body>)
  - Imported by view JSPs using the <wcpgl:widgetImport> tag
- Stores.war/Widgets/<widgetidentifier>/<widgetname>_UI.jspf
  - Widget UI JSPF
  - Formats data fetched by widget data JSPF for presentation
  - Some widgets can have multiple widget UI JSPFs for different scenarios

Store assets (cont'd)

- Stores.war/<storedir>/css/base.css
  - Stylesheet for responsive CSS rules common to all store pages
  - Contains responsive grid system
  - Imported by:
    - Stores.war/<storedir>/css/common1_1.css
    - Stores.war/<storedir>/mobile30/css/common1_1.css
- Stores.war/<storedir>/css/styles.css
  - Stylesheet linked by the responsive view JSPs
- Stores.war/<storedir>/css/common1_1.css
  - Stylesheet linked by all view JSPs
- Stores.war/<storedir>/mobile30/css/common1_1.css
  - Stylesheet linked by the mobile view JSPs

Customization

Creating responsive layout templates

- Refer to the Commerce Composer session on how to create a layout template
- The simplest way to make a layout template responsive is by using a responsive grid system, such as the one we ship OOTB
- How to use a responsive grid system to markup a container JSP:
  - Divide the container into rows and columns using the CSS classes provided by the grid system
  - Markup the slots using the <wcpgl:widgetImport> tag
  - Add responsive behaviors (such as column stacking) to the slots using the CSS classes provided by the grid system
  - Customize the responsive behaviors of the slots using media queries
- Preview and test against different widgets and viewport sizes!

Creating responsive widgets

- Refer to the Commerce Composer session on how to create a widget
- Do NOT fix the width of the widget or preset its margin. i.e. the root HTML element of the widget should have the following CSS styles:
  - display: block; (default)
  - width: auto; (default)
  - Margin: 0; (default)
- This will allow the layout/container/slot to determine its width and the spacings between widgets.
- Contrary to width, you can give the widget a fixed height or minimum height, or let its content determine its height (i.e. height: auto) - the latter is better if you're making the widget fluid

Creating repsonsive widgets (cont'd)

- Make the widget's internal UI elements fluid using CSS. Some common techniques:
  - Use a responsive grid system to arrange the UI elements into rows and columns, such as the one we ship OOTB
  - Use display: inline-block or float: left to flow the UI elements left-to-right, top-to-bottom
  - Use overflow: scroll in combination with min-width if you need to display UI elements that requires a minimum amount of horizontal space (e.g. tables)
  - Avoid using absolute positioning in a fluid container unless you're trying to anchor the UI element to a specific corner or side of its container
- Text blocks:
  - Avoid fixing the heights of text blocks in a fluid container since text might reflow/rewrap when the container's width changes. If you need the text block to have a minimum height use min-height
  - Avoid using line-height to specify the height of a single-line text block - it won't look good if the text needs to wrap. Use padding instead if you need to pad the text block
  - Consider using display: table-cell if you need to vertically center text that may wrap

Creating responsive widgets (cont'd)

- If possible, optimize the widget's UI elements further for different viewport/slot widths:
  - If you're using a responsive grid system (like ours), use it to define responsive behavior like column stacking on smaller screens
  - Use media queries to optimize the widget's styling for different breakpoints
- Use JS to split the widget's content into multiple pages depending on the amount of space available (similar to our recommendation widgets)
  - Be aware that users on touch screens will expect pagination to work via swiping so if possible, do implement touch support
- Preview and test against different layout templates and viewport sizes!

Creating responsive marketing contents

- Responsive marketing contents are marketing contents with live image and text layers that, unlike flat images, can adapt to different viewport widths
- General guidelines:
  - Use HTML to layer the images and text instead of flattening them into a single image
  - Position and scale images using CSS
  - Use media queries to conditionally load smaller images on smartphones
  - Adjust font size using media queries and allow text blocks to flow naturally
- Alternatives:
  - Create device-specific layouts with different marketing contents for different devices
  - Our content recommendation widget will scale non-responsive marketing contents (e.g. flat images) to fit the slot width (to a maximum of 100%)
  - Use 3rd party client and server side solutions* that will fetch or serve images and videos based on viewport width or the browser's user agent

*Examples of such solutions:

- Adaptive images: http://adaptive-images.com/
- Img srcset polyfill: https://github.com/borismus/srcset-polyfill

Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, Coremetrics, DB2, PowerVM, Rational, WebSphere, and z/VM are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, 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 PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION 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 ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.