XPagesExtLibX

This project is an OpenNTF project, and is available under the Apache Licence V2.0. All other aspects of the project, including contributions, defect reports, discussions, feature requests and reviews are subject to the OpenNTF Terms of Use - available at http://openntf.org/Internal/home.nsf/dx/Terms_of_Use.

The source code of the project is available under the Apache License V2.0 at: https://github.com/OpenNTF/XPagesExtLibX.

Bootstrap 4 Alpha Introduction

"Bootstrap is the most popular HTML, CSS, and JS framework in the world for building responsive, mobile-first projects on the web." – *Bootstrap website*

Bootstrap 4 is the next major release of the open-source Bootstrap project. The first release was over 4 years ago, and each major release has brought with it significant changes. Currently, Bootstrap 4 is in alpha state, with a yet to be announced full release date. But we are adding the alpha version of Bootstrap 4 to the ExtLibX project now in order to start preparing for Bootstrap 4's future release. By laying the foundation for Boostrap 4 in XPages in the ExtLibX project during the alpha phase, it will facilitate its addition to the core Extension Library in a prompt manner once Bootstrap 4 is actually released. Futhermore, it gives the XPages community the opportunity to contribute to the development of Bootstrap 4 support in XPages between now and its official release.

Notable Changes and Additions in Bootstrap 4

This section outlines some of the significant changes coming in the new version of Bootstrap, and how they may affect existing Bootstrap support in XPages.

- "rem"s – All typography sizes are now based on rem (root em) , 'px' and 'pt' are no longer used in the Bootstrap CSS for fonts. This means that all font sizes throughout Bootstrap 4 are set as a fraction of the font-size of the root html element. The default root font-size is 16px. This means that if you stick with that default value and set "font-size: 1rem" on an element, it will also be given a font-size of 16px. But you can scale up or down the font-size for individual elements, using other values such as "font-size: 2rem" (32px by default), "font-size: 0.5rem" (8px by default) and so on. This also means that if you change the font-size on the root html element, all font-sizes throughout your application will change accordingly. Thus it is very easy to adjust your application for different devices. The grid system of Bootstrap is also now based on rems, so one change to the font-size on the root html element will scale your entire application up or down. This makes life very easy when tailoring your application for different devices. In the XPages renderers and CSS, we have also adopted this approach, using rems wherever possible.
- Updated media queries – The media-queries in Bootstrap 4 now use 'em's instead of pixels, and they have 5 breakpoints for devices. Extra small (< 34em), small (< 48em), medium (< 62em), large (< 75em), and extra large (> 75em). The extra large size is new in Bootstrap 4, and comes with associated CSS classes for various components, using 'xl' to reference that

size. For example ‘col-xl-4’, which specifies the grid column width of an element when using an extra large device.

- Cards – Cards have become a popular container element in recent time, and they are often added on top of Bootstrap 3 by those using it. Now Bootstrap 4 brings Cards into its feature set by default. They replace panels and wells from Bootstrap 3, providing a “flexible and extensible content container”. They come with customisable headers, footers & backgrounds, wrapping any content you wish to contain in them. And you can use Card Groups or Card Decks to group cards together. All in all a powerful new component that you can use to build your applications.

- Navbars refactored – The navbar component has been re-worked in Bootstrap 4. There have been changes to the structure, the CSS classes used, with new classes added, others removed and the underlying CSS properties updated. This is quite a significant change, as any navbar written for Bootstrap 3 will be completely broken when switching to Bootstrap 4. It is also a component heavily used in XPages, with both a dedicated navbar control and integrated navbars used in the application layout control. This is one of the first things we have tackled in the first release of the BootstrapX plugin in ExtLibX.

- Button classes – The ‘.btn-default’ CSS class has been removed in Bootstrap 4. A default button only needs the ‘.btn’ CSS class instead of ‘.btn .btn-default’ from older Bootstrap versions. There’s a new ‘btn-secondary’ class, to compliment the ‘.btn-primary’ class. And there are new outline buttons using classes of the format ‘.btn-XXX-outline’, where ‘XXX’ is replaced by ‘primary’, ‘secondary’ or ‘success’ etc.

- Glyphicons dropped – The glyphicons that were included in Bootstrap 3 have been left out of Bootstrap 4. To use any icon font, app devs would need to pull the resources into their applications themselves. However, the XPages team has decided to maintain a Bootstrap 3 version and a Bootstrap 4 version in the Extlib, in line with Bootstrap’s own support plan. Therefore, we’ve included the glyphicons from Bootstrap 3 as a resource in the Bootstrap 4 theme. This makes for a simpler transition from Bootstrap 3 themes to the Bootstrap 4 theme.

- IE8 support dropped – IE8 support in Bootstrap 3 was semi-supported, as it still required additional resources in order to work, e.g. the Respond.js project. In Bootstrap 4 though, support for the legacy browser is being dropped completely.

- Framework changes – There have been changes to the way that Bootstrap 4 is built. It uses SASS instead of LESS, the JS is entirely written in ES6 and generally it is easier to create your own customised version of Bootstrap. For XPages though, we are adding the default full version of Bootstrap 4 and build upon that with our renderers and theme(s) to provide Bootstrap 4 support in ExtLibX.

- Smaller footprint – Bootstrap 4 is roughly 30% smaller than its predecessor, which is a significant improvement. Every kilobyte counts when it comes to application performance, especially on mobile devices.

There are many more changes in Bootstrap 4, too many to cover here, but you can read more in the Bootstrap 4 blog http://blog.getbootstrap.com/, and in the Bootstrap 4 documentation http://v4-alpha.getbootstrap.com/getting-started/introduction/

List of work done and work that needs to be done

Thus far the work in this project has used the Todo.nsf, which is part of recent Extension Library releases, as the basis for initial work in integrating Bootstrap 4 alpha into XPages. Since this is a basic responsive application using Bootstrap heavily, it is a useful test case for the migration to Bootstrap 4. The application should now look and function well using the Bootstrap 4 theme, as well as it does when using Bootstrap 3.

The list of work done so far to support Bootstrap 4 in XPages:

- New Bootstrap4 plugin added to ExtLibX project.
- Bootstrap 4 resources added to the new plugin.
- Bootstrap 4 theme added to the new plugin. This theme incorporates all the Bootstrap 4 resources & applies new Bootstrap 4 renderers. It also points to jQuery & glyphicons from the existing com.ibm.xsp.theme.bootstrap plugin in the ExtLib.
- XPages specific CSS & JS files added.
    - Glyphicon classes added to xspBootstrap4.css.
    - Font sizes replaced with rems.
    - Primary colour updated to '#0275d8'.
    - Date Time Picker CSS cleaned up.
    - CSS cleaned up for some other issues/controls
- Application Layout renderers implemented when using either the SimpleResponsiveApplicationConfiguration or the ResponsiveApplicationConfiguration.
    - Associated renderers for elements of the Application Layout have been implemented, such as the PlaceBarActionsRenderer & ApplicationLinksRenderer.
- Navbar control renderer implemented.
- Dashboard control renderer implemented.
- Navigator control renderer implemented.
- Data View control renderer implemented.

Todo items:

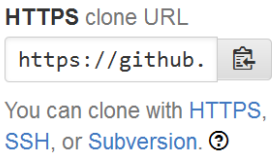
- Fix JUnit failures in ExtLibX JUnit tests.
- All other XPages controls not listed in the Done section above, need to be tested and have any necessary Bootstrap 4 renderers or CSS tweaks implemented.
- The responsive behaviour for navbar & application layout needs more testing/tweaking.
- The responsive behaviour of the dashboard needs more testing/tweaking.

- Changes to the SimpleResponsiveLayoutConfiguration class in the com.ibm.xsp.theme.bootstrap plugin need to be delivered to the core Extlib to support the new SimpleLayoutRenderer in the BootstrapX plugin.

Creating your fork of the ExtLibX repository

Follow the steps below to create your own personal copy of the ExtLibX project code so you can begin to work on it and contribute back.

1. Create your personal "Fork" of the ExtLibX repository from the "Fork" button on the project homepage -> https://github.com/OpenNTF/XPagesExtLibX
2. On Github, navigate to your fork of the ExtLibX repository. https://github.com/UserName/XPagesExtLibX
3. In the right side of your fork's repository page click  to copy the clone URL for your fork



4. Open Terminal (for Mac and Linux users) or command prompt (for Windows users)
5. Type "git clone" then paste the URL you copied in step 2
6. Press Enter and your local clone will be created

Next you will want to use your Integrated Development environment of choice to import the plugins and feature to start to run and debug the ExtLibX library.

There are many tutorials on how to contribute code to a project via GitHub.

Some examples are:

https://guides.github.com/activities/contributing-to-open-source/

http://kbroman.org/github_tutorial/pages/fork.html

Also here is a list of related articles and information on setting up a Domino OSGi development environment from IBM'ers and community members.

http://www14.software.ibm.com/cgi-bin/weblap/lap.pl?l=en&li_formnum=L-GHUS-89PP6N&popup=n

http://www.openntf.org/main.nsf/blog.xsp?permaLink=NHEF-8R9AMD

http://developmentblog.johnmcooper.co.uk/2014/05/configuring-eclipse-for-xpages-osgi-plugins-part1.html

http://www.slideshare.net/fiorep/domino-osgi-development

http://www.gregorbyte.com/2014/09/domino-debug-plugin-fix-for-eclipse-luna.html

Making changes in ExtlibX to support Bootstrap 4

This section covers the basic steps required for XPages community members to contribute code to Bootstrap4 support in the ExtLibX github project.

In order to properly support and utilise Bootstrap 4, XPages controls will often require a new renderer to be implemented. This is a common occurrence when implementing new themes. However, in some cases the existing renderer in the com.ibm.xsp.theme.bootstrap plugin of the ExtLib is sufficient, or perhaps only minor CSS changes are necessary. Below we describe the processes for both situations, when only CSS changes are required, and when a new rendererer is required.

1. CSS changes

   In the event that an XPages control simply needs some CSS changes to be used with Bootstrap 4, there are two CSS files that are relevant.

   a. xspCoreBootstrap4.css – This file is used for any CSS pertaining to the core controls of XPages for usage with Bootstrap4. That is, any control referenced using the core library of XPages, xmlns:xp="http://www.ibm.com/xsp/*core*", e.g. “<xp:panel>”.

   b. xspBootstrap4.css - This file is used for any CSS pertaining to the ExtLib controls of XPages for usage with Bootstrap4. That is, any control referenced using the extentsion library of XPages, xmlns:xe="http://www.ibm.com/xsp/coreex", e.g. “<xe:dataView>”.

   Search the relevant CSS file for any CSS already pertaining to the control in question, and add the necessary changes alongside any existing CSS for that control. Or if no CSS related to the control is in the file, add it with a commented heading that names the control to which the CSS applies. For example:

```
/** Navbar styles **/
.xspNavItem {
     padding: .425rem 0rem;
}
```

2. Adding a renderer

   Often it will be necessary to add a renderer for a control, there have already been a number of renderers added for Bootstrap 4 support (the main additions were listed earlier in this documentation). To see the simplest possible example of one of these renderers, look at the file:

   com.ibm.xsp.extlibx.theme.bootstrap4.renderkit.html.extlibx.data.DataViewRenderer

   This renderer simply overrides one method from the super-class: com.ibm.xsp.theme.bootstrap.renderkit.html.extlibx.data.DataViewRenderer. That is the renderer from the existing com.ibm.xsp.theme.bootstrap plugin in the ExtLib (i.e. a Bootstrap 3 renderer) for the Data View control.

   The overridden method getProperty is a common method amongst renderers in the Extension Library. It allows certain key renderer variables to be altered on a per renderer

basis, simply by overriding the getProperty method and changing the value returned for specific variables. This can save you from overriding and re-implementing more complex renderer methods just because you need to make a small change to the version in the super-class renderer.

In the case of the new DataViewRenderer, it is changing one specific property, the 'PROP_SUMMARYTITLETAG' property. This property defines the html tag that encloses the summary text for an entry in the data view. A few super-classes up, it is set to 'h4', whereas for Bootstrap 4 we are changing it to 'h6'. With the new renderer, it will output html as follows for the summary text of a row in the data view:

```
<div>
	<h6 id="…" class="xspDataViewSummary" style="margin: 0">
		<a id="…" href="doc.xsp">test1</a>
	</h6>
</div>
```

Once you have implemented your renderer, you need to take two more steps before it gets used.

1. Add the renderer to the faces-config.xml file of the plugin (com.ibm.xsp.extlibx.theme.bootstrap4.config.extlibx-bootstrap4-faces-config.xml). For the renderer, you need to define a component-family, a renderer-type and a renderer-class.

```
<renderer>
<component-family>com.ibm.xsp.DataIterator</component-family>
<renderer-type>com.ibm.xsp.extlibx.theme.bootstrap4.data.DataView</renderer-type>
<renderer-
class>com.ibm.xsp.extlibx.theme.bootstrap4.renderkit.html.extlibx.data.DataViewRend
erer</renderer-class>
</renderer>
```

The component-class should match the component-class associated with the super-class renderer, in the super-class' corresponding faces-config file. The renderer-type needs to be different to the super-class' renderer-type, so that it is unique. The renderer-class references the class you have implemented for the new renderer, e.g. com.ibm.xsp.extlibx.theme.bootstrap4.renderkit.html.extlibx.data.DataViewRenderer.

2. With the renderer defined in the faces-config.xml file, you can now reference it in the theme file. In Bootstrap4.theme, you can see the example for the DataView renderer:

```
<control>
	<name>DataIterator.DataView</name>
	<property>
		<name>rendererType</name>
		<value>com.ibm.xsp.extlibx.theme.bootstrap4.data.DataView</value>
	</property>
</control>
```

You need to set the rendererType property for the control to match the renderer-type you specified in the faces-config.xml.

With the renderer implemented, added to the faces-config and specified in the theme file, that new renderer will be used to render that control in applications using the Bootstrap4 theme.

In some cases, you will need more detailed renderer code to override the super-class. This could be because the new rendering you need is very different to the previous implementation. Or because the super-class doesn't have the necessay extensible property variables available that you can override, like we did in the DataView renderer. In the second case, it could be worth making changes to the super class renderer to support more property variables, and submit a pull request to the ExtLib github repository. This would mean less code needs to be written in the new renderer for Bootstrap4, keeping things simpler.

There may also arise situations where you need to implement a completely new renderer from scratch. In such cases your new renderer should extend com.ibm.xsp.extlib.renderkit.html_extended.FacesRendererEx, a basic XPages ExtLib renderer class.

Of course, if you find any problems with the renderers already implemented, you should feel free to find a fix for the problem(s) and submit a pull request to the git repo.