[EN] 04. Adjusting the Look of the Reader Application

Adjusting the Look of the Reader Application

Adding and Modifying Content in a Component

Most data displayed by the reader application is downloaded from the dLibra system server. Some examples of such data are collections, digital objects, or attribute values. However, the user's own, non-standard content can be added, and the manner in which the existing content is displayed can be changed in any place of the displayed HTML code.

Example: We want to place an image – which presents the employees of the institution – on the "Contact" page.

1. We open the "Contact" page and verify the name of the page in the address bar – for example, for the https://example.com/dlibra/contact?language=pl address, the page name in the reader application will be "**contact**".
2. We open the **/WEB-INF/pages.xml** file and find the <page name="**contact**" layout="**home_layout**" secure="true" inherits="**base-page**"> line. In that section, there are all the available components for the "contact" page. We are interested in the only **ContactComponent** component. Apart from its components, the **contact** section uses components from the **base-page** section (but we do not want to modify it in any way). The section for the home page – for example, https://example.com/dlibra/ – is here called "**main**" (we do not want to modify it, either). For making other changes, it may be useful to know that the "contact" page is embedded in the **home_layout** template. We can introduce changes in that template as well. We will find it in the /WEB-INF/layout/templates directory. Before a modification, a backup of the directory should be made.
3. We unpack the **/WEB-INF/lib/dcore-webapp-components-6.0.0.jar** archive. Having done that, we copy the **ContactComponent.vm** file to the **/WEB-INF/components/templates** directory.
4. From that time on, we can make any changes in the **ContactComponent.vm** file. The changes will have an impact on the content displayed on the "Contact" page. When the **ContactComponent.vm** has been removed from the **/WEB-INF/components/templates** directory, our changes will disappear, and the default manner of displaying content will be restored.
5. In the **/WEB-INF/components/templates** file, we search for a place in which we would like to add the image and add the appropriate HTML fragment, for example:

Adding a New Component to a Subpage

If we decide that we want to add our own data, static or downloaded with the use of a remote API, we can create our own component.

Example: We want to display weather data on the home page, for English-speaking guests.

1. In the **/WEB-INF/components/templates** , we create a **WeatherComponent.vm** file, with the following content.

WeatherComponent.vm

```
<div #if($userLanguage != "en") style="display:none" #end id="WeatherComponent"></div>
```

2. We create the **/style/dlibra/custom/js**, directory containing the **WeatherComponent.js** js file with the following content (the script downloads the weather data from an external Yahoo server and displays them on our page, in the selected place).

WeatherComponent.js

```
var dlibra = dlibra || {};
dlibra.WeatherComponent = {
        init: function () {
                $.get( "https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20weather.forecast%
20where%20woeid%20in%20(select%20woeid%20from%20geo.places(1)%20where%20text%3D%22poznan%2C%20pl%22)
&format=json&env=store%3A%2F%2Fdatatables.org%2Falltableswithkeys", function( data ) {
                  var description = data.query.results.channel.item.description.replace("<![CDATA[", "").
replace("]]>", "");
                  $('#WeatherComponent').html(description)
                });
        }
}
$(function () {
        dlibra.WeatherComponent.init();
})
```

3. We edit the **/WEB-INF/layout/templates/home_layout.vm** file and, just before closing the “body” tag, add our script.

home_layout.vm

```
        <script src="${homepageUrl}/style/dlibra/custom/js/WeatherComponent.js" type="text/javascript"><
/script>
</body>
```

4. We add the **WeatherComponent** component to the home page. For that purpose, we look for the “main” section which corresponds to the home page, in the **/WEB-INF/pages.xml** file. We add another component inside components.

pages.xml

```
<page name="main" layout="home_layout" inherits="base-page">
                  <actions>
                          <action name="ChangeLanguageAction"/>
                          <action name="LogoutAction"/>
                  </actions>
                <components>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.
LatestEditionsComponent">
                                <place>topSplit</place>
                                <position>1</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.NewsComponent">
                                <place>topSplit</place>
                                <position>2</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.WeatherComponent">
                                <place>main</place>
                                <position>1</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.
CollectionsPresentationComponent">
                                <place>main</place>
                                <position>2</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.
PopularEditionsComponent">
                                <place>main</place>
                                <position>3</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.RecommendedComponent">
                                <place>main</place>
                                <position>4</position>
                        </component>
                </components>
</page>
```

We ought to remember to assign a higher position to the remaining components the place of which is also "**main**", so that every component has a different **position**.

5. The system will display the weather information for English-speaking guests on our site.

Przykładowy komponent pogody

Removing a Component from a Subpage

For various reasons, a digital library administrator may decide to remove some components. There are two methods of removing a component:

- by commenting it out in file **/WEB-INF/pages.xml** (this is a safer method because it allows the administrator to restore the component) or
- by removing the component from file **/WEB-INF/pages.xml**

Example: We decide to hide the component with the most popular objects displayed on the home page.

1. We open the **/WEB-INF/pages.xml** file and look for the **main** section which contains the components displayed on the home page. We comment the **PopularEditionsComponent** component with the <!— <component>...</component> —> comment, as shown in the file below.

pages.xml

```
        <page name="main" layout="home_layout" inherits="base-page">
                  <actions>
                          <action name="ChangeLanguageAction"/>
                          <action name="LogoutAction"/>
                  </actions>
                <components>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.
LatestEditionsComponent">
                                <place>topSplit</place>
                                <position>1</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.NewsComponent">
                                <place>topSplit</place>
                                <position>2</position>
                        </component>
                        <component name="pl.psnc.dlibra.web.comp.pages.components.
CollectionsPresentationComponent">
                                <place>main</place>
                                <position>1</position>
                        </component>
                        <!--<component name="pl.psnc.dlibra.web.comp.pages.components.
PopularEditionsComponent">
                                <place>main</place>
                                <position>3</position>
                        </component>-->
                        <component name="pl.psnc.dlibra.web.comp.pages.components.RecommendedComponent">
                                <place>main</place>
                                <position>4</position>
                        </component>
                </components>
          </page>
```

Setting a Component in a Layout

Every component has its place in the page layout. The information about which place exactly the component is located in can be checked in the pages.xml file. In that file, we can also indicate a different place for a file.

Example: We decide to move the news component to a completely different place on the home page.

1. We open the **pages.xml** file.
2. We look for the **main**, section, which contains the components from the home page.
3. We read the name of the **layout**, from the layout field. The default layout name for a page is **home_layout**.
4. In the **main** section, we look for the component with updates, **pl.psnc.dlibra.web.comp.pages.components.NewsComponent**, to which we want to assign a new location in the **home_layout** layout. We read the place for that component from the place field. By default, the value is **topSplit**. We check the order in the **position**; by default, the value is **2**. The first component displayed in the topSplit place is the component with recently added objects.
5. We modify the value of **field** place in the data of component **pl.psnc.dlibra.web.comp.pages.components.NewsComponent**. We change value **place** to **myPlace**, and value **position** – to **1**.Każdy komponent ma swoje ściśle określone miejsce w layoucie strony. Informację o tym, w którym konkretnie miejscu layoutu znajduje się dany komponent możemy sprawdzić w pliku **pages.xml**. Tam możemy również wskazać inne miejsce wyświetlania się komponentu.

pages.xml

```
<component name="pl.psnc.dlibra.web.comp.pages.components.NewsComponent">
        <place>myPlace</place>
        <position>1</position>
</component>
```

6. We open file **/WEB-INF//layout/templates/home_layout.vm** and place the following code anywhere in that file.

home_layout.vm

```
#foreach( $comp in $myPlace)
        $!{comp.RenderedTemplate}
#end
```

As a result, all components the **place** fields of which have value **myPlace** will be displayed, including our component with updates

Modifying and Adding Language Messages

Every message in the reader application is displayed in multilingual form. It means the language of a displayed message will differ depending on the selected website language.

Example: We want to the footer to include the “All rights reserved” clause.

1. We look for the file with the information which is displayed at the very bottom of a page. By analyzing file **/WEB-INF/layout/templates /home_layout.vm**, we will learn that the footer content is determined by the **/WEB-INF/layout/templates/parts/footer.vm** file.
2. We add a new entry in the **/WEB-INF/layout/templates/parts/footer.vm** file.

   footer.vm

   ```
   <span class="footer__bottom-text">$res.get('Home.Copyright.clause')</span>
   ```

   The **$res.get('Home.Copyright.clause')** reference will display the clause in various languages. They must be defined.

3. We open the **/WEB-INF/layout/resources/layout_pl.xml** file and add a translation for Polish.

layout_pl.xml

```
<entry key="Home.Copyright.clause">Wszelkie prawa zastrzeone</entry>
```

4. We open the **/WEB-INF/layout/resources/layout_en.xml** file and add a translation for English.

layout_en.xml

```
<entry key="Home.Copyright.clause">All rights reserved</entry>
```

5. We save the messages we would like to display in components in files **/WEB-INF/components/resources/WEBAPP_{język}.xml**. Next, in component files (for example, **MyComponent.vm**), we refer to translations in the following way:

MyComponent.vm

```
$res.get('MyComponent.MyInfo')
```

Changing the Existing Images (Logo)

Apart from dynamic images (for example, digital object thumbnails), the reader application displays static images (for example, logo). The second group of images can be changed to user's own. It is recommended that the new images be the same size as the replaced images. If the sizes of a new image is different, the display of that image should be tested in various views and browsers, for a desktop computer, a tablet, and a mobile phone.

Example: We change the default dLibra logo to our own.

1. We open the **/WEB-INF/layout/custom_templates/custom_layout_library.vm** file and add an entry with the path to the new logo:

```
#macro(logoPath)
        ${homepageUrl}/style/dlibra/${styleVariant}/my/logo/path/logo.svg
#end
```

Example: We change the default dLibra logo to our own, taking into account the language version of the logo.

1. We open the **/WEB-INF/layout/custom_templates/custom_layout_library.vm** file and add an entry with the path to the new logo. Using the ${userLanguage} variable, we takes into account the logos in various languages:

```
#macro(logoPath)
        #if ( ${userLanguage} == "en" )
                  ${homepageUrl}/style/dlibra/${styleVariant}/my/logo/path/logo_en.svg
        #elseif ( ${userLanguage} == "de" )
            ${homepageUrl}/style/dlibra/${styleVariant}/my/logo/path/logo_de.svg
        #else
                ${homepageUrl}/style/dlibra/${styleVariant}/my/logo/path/logo_pl.svg
        #end
#end
```

Example: We change an image to our own.

1. In order to change an image, we just have to find the path of the old image and paste the new one there. The best way to check the path of an image is to use the **Google Chrome** browser.
2. We open the page we are interested in in the digital library. We right-click the image and choose the examine option from the context menu. There will appear a window with the element selected in the HTML code. The element will be **img**. We check the src value of the element to read the path to the file on our server. For example, value src="**http://demo.dl.psnc.pl/dlibra/default/img/pictures/my-picture.svg**" points to directory **/webapp/style/dlibra/default/img/pictures/my-picture.svg.**

Modifying CSS

The styles used by the reader application are in the **/styles/dlibra/default/css/** directory. They can be overwritten, and new styles can be added. The most appropriate place for it is file **/styles/dlibra/default/css/custom.css**.