Creating themes for LibreCCM with Freemarker

Starting with version 2.5 the LibreCCM platform support Freemarker as an alternative to XSL. Freemarker is a project of the Apache Foundation and a well known and mature template engine for Java. The support for Freemarker in version 2.5 is a backport from the upcoming version 7 of the LibreCCM platform.

Compared to XSL Freemarker is a lot easier to use, especially if you have worked with other template engines like Twig, Velocity etc before. In version 7 of the LibreCCM platform Freemarker will be become the primary template engine. XSL will still be supported, but we recommanded that you port your themes to Freemarker. Why Freemarker and not one of the other template engines? Freemarker is able to process XML in a similar way than XSL.

Freemarker also allows it to define user defined directivies and functions. To make it easier to create impressive themes we provide functions and macros for Freemarker we provide several functions and macros which hide the complexity of the XML data model created by CCM from the template author. It is recommanded not to access the XML data model directly. Instead the provided functions should be used. Otherwise your theme might brake when the XML structure changes.

General structure of a Freemarker theme

Freemarker themes have a different structure than the usual “old style” themes of LibreCCM.

ToDo

Predefined variables and functions

Several variables and functions are predefined and available without importing another file.

Variables

`contextPath`

The context path in which CCM is running.

`contextPrefix`

The context prefix.

`dispatcherPrefix`

Prefix for the CCM dispatcher (usually `/ccm`)

`host`

The current host.

`model`

The XML document created by LibreCCM.

`negotiatedLanguage`

The language negoiated between the user agent and LibreCCM.

`requestScheme`

The protocol (http or https).

`selectedLanguage`

The language selected by the user.

`themePrefix`

The prefix of the theme. Only available if a development theme is viewed.

Functions

`getLocalizedText`

```
String getLocalizedText(String key)
```

Returns the localized text from the resource bundle of the theme.

`getContentItemTemplate`

```
String getContentItemTemplate(String objectType, view="DETAIL", style="")
```

This is an internal function!

Returns the path for the template of a content item of a specific type.

`_formatDateTime`

An internal functions date time formatting. This functions should not be used directly.

Functions and Macros

Common functions

Language related

Import path: `<#import /language.ftl as Lang>`

`getAvailableLanguages`

`Sequence getAvailableLanguages()`

Returns the available languages for the current document as sequence. These sequence can be used for creating links for selecting the language:

```
<ul class="language-selector">
    <#list Lang.getAvailableLanguages()?sort as lang>
        <li class="${(lang==negotiatedLanguage)?then('selected', '')}">${lang}</li>
    </#list>
</ul>
```

This example uses the `list` directive from Freemarker to iterate over the available languages returned by `getAvailableLanguages` The Freemarker build-in `?then` is used together with the `negotiatedLanguage` variable to check if the curent language is the selected language. If this is the case a CSS class is added to the HTML.

Basic functions

Import path: `<#import /utils.ftl as Utils>`

getPageApplication

`getPageApplication()`

Return the application of the current page.

getPageTitle

`getPageTitle()`

Returns the title of the current page

getSiteHostName

`getSiteHostName()`

Returns the name of the host serving the site.

getSiteName

`getSiteName()`

Returns the name of the site.

getBooleanAttrValue

`getBooleanAttrValue(fromNode: Node attrName: String)`

A helper function which tries to convert the value of the attribute `attrName` of the node `fromNode` to a boolean. The following values are interpreted as true: `true`, `yes`. All other values are interpreted as `false`.

Parameters

`fromNode` A XML node

`attrName` The name of attribute to interpret as boolean

Returns

A boolean for the value of the attribute. If the attribute is not present in the provided node the function returns `false`.

formatDateTime

`formatDateTime(style: String date: Node)`

Formats the value of date/time value node according to the provided `style`. The is defined in the theme manifest in the `date-time-formats` section. It is possible to define different styles for different languages. The style definition in the theme manifest must be in the format expected by the Java DateTimeFormatter class.

Parameters

`style` A date-time format defined in the theme manifest. The format must be formatted as expected by the DateTimeFormatter class.

`date` The node providing the data of the date to format.

Returns

A date formatted as defined in `style`.

Examples

In the theme manifest in the following format is defined:

```
"date-time-formats": [
    ...
    {
        "style": "news",
        "lang": "de",
        "format": "dd.MM.YYYY"
    },
    {
        "style": "news",
        "lang": "en",
        "format": "MM/dd/YY"
    },
    ...
]
```

The use this format:

```
Utils.formatDateTime('news', News.getDateTime(item))
```

`News.getDateTime` gets the date of a news item. If the date of the news is 2019-04-01 the return value of the function for german is

```
01.04.2019
```

and for english:

```
4/1/19
```

ccm-cms-assets-fileattachment

This module provides functions for dealing with file attachments. A possible usage these functions:

```
<#list FileAttachments.getFileAttachments(item)>
    <div class="file-attachments">

        <h2>
            ${getLocalizedText("layout.page.main.fileAttachments")}
        </h2>

        <ul class="file-attachments">
            <#items as file>
                <#if FileAttachments.getFileType(file) == "caption">
                    <li class="caption">
                    <strong>${FileAttachments.getFileName(file)}</strong>
                    <p>
                        ${FileAttachments.getFileDescription(file)}
                    </p>
```

```
                    </li>
                <#else>
                    <li class="file-attachment">
                        <a href="${FileAttachments.getFileUrl(file)}">
                            <span class="fa fa-download"></span>
                            ${FileAttachments.getFileDescription(file)}
                            (${FileAttachments.getMimeTypeFileExtension(file)},
                            ${FileAttachments.getFileSize(file, "KiB")} KB)

                        </a>
                    </li>
                </#if>
            </#items>
        </ul>
    </div>
</#list>
```

getFileAttachments

`getFileAttachments(item: ContentItemNode): Sequence`

Return the file attachments of a content items

Parameters

`item` The content item from which the file attachments are retrieved.

Returns

A sequence of the file attachments of the provided content item.

getFileType

`getFileType(file)`

Returns the type of the file attachments which is either `caption` or `file`.

Parameters

`file` The file attachment

Returns

The type of the file attachment.

getMimeType

`getMimeType(file)`

Returns the mime type of the file.

Parameters

`file` The file

Returns

The mime type of the file.

getMimeTypeFileExtension

`getMimeTypeFileExtension(file)`

Returns the usual file extension for the mime type of the file.

Parameters

`file` The file

Returns

The usual file extension for the mime type of the file.

getFileSize

`getFileSize(file unit="byte")`

Returns the size of the file in the provided unit.

Parameters

`file`The file

`unit` Optional parameter for unit in which the size is returned. Default value is `byte`. Supported values are `byte`, `kB` `KiB`, `MB` and `MiB`. All other values are interpreted as `byte`.

Returns

The size of the file in the provided unit.

getFileId

`getFileId(file)`

Returns the ID of the file.

Parameters

`file` The file

Returns

The ID of the file.

getFileName

`getFileName(file)`

Returns the name of file.

Parameters

`file` The file

Returns

The name of the file.

getFileDescription

`getFileDescription(file)`

Returns the name of file.

Parameters

`file` The file

Returns

The description of the file.

getFileUrl

`getFileUrl(file)`

Returns the name of file.

Parameters

`file` The file

Returns

The URL of the file.

ccm-cms-assets-imagestep

Provides functions for dealing with image attachments of a content item.

Import path: `<#import "/ccm-cms-assets-imagestep.ftl" as Images>`

Example usage:

```
<#import "/ccm-cms-assets-imagestep.ftl" as Images>

<#list Images.getImageAttachments(item)>
    <div class="image-attachments">
        <#items as image>
            <figure>
                <div>
                    <a data-fancybox="gallery">
                        <img src="${Images.getImageUrl(image)}"
                             width="100%"
                             height="auto" />
                    </a>
                </div>
                <figcaption>
                    ${Images.getImageCaption(image)}
                </figcaption>
            </figure>
        </#items>
    </div>
</#list>
```

getImageAttachments

`getImageAttachments(item)`

Get the image attachments of a content item

Parameters

`item` The content item.

Returns

A sequence of the image attachments of the provided content item.

getImageId

`getImageId(image)`

Get the ID of the provided image.

Parameters

`image` The image.

Returns

The id of the image.

getImageName

`getImageName(image)`

Gets the name of the provided image.

Parameters

`image` The image.

Returns

The name of the image.

getImageCaption

`getImageCaption(image)`

Gets the caption of the provided image.

Parameters

`image` The image.

Returns

The caption of the image.

getImageSortKey

`getImageSortKey(image)`

Gets the sort key of the provided image.

Parameters

`image` The image.

Returns

The sort key of the provided image.

getImageWidth

`getImageWidth(image)`

Gets the width of the provided image.

Parameters

`image` The image.

Returns

The width of the provided image.

getImageHeight

`getImageHeight(image)`

Gets the height of the provided image.

Parameters

`image` The image.

Returns

The height of the provided image.

getImageUrl

`getImageUrl(image)`

Gets the URL of the provided image.

Parameters

`image` The image.

Returns

The URL of the provided image.