Apache OFBiz Themes

The Apache OFBiz Project

Version unspecified, Release 17.12

Table of Contents

1. What is a theme

A Theme is an ofbiz component that defines all elements necessary to render all information generated by the screen engine through an embedded technology. Currently themes presents in Apache OFBiz use html5/jquery/css to do that.

To realize that, a theme can define some properties, among them some can be necessary. It can define its own ftl macro to render the different modelScreen elements and can define its own screen decorator to prepare the final user screen structure for the technology used by the theme.

2. How to define it

A theme is a standard component, present in the directory themes or plugins with a file definition present on widget/Theme.xml and support the widget-theme.xsd

To offer the possibility for end users to select the theme through the "Select Theme Screen", the theme need to load the entity VisualTheme and one or more entries related to this theme.

3. Structure of Theme.xml

The theme definition file help OFBiz to know what specific rendering the Theme want to use.
It’s composed by two definition elements and four optional blocks

3.1. Main definition

The First mandatory element is one or several visualThemeId related to the theme
Defined like that

```
    <visual-themes>
        <visual-theme id="MY_THEME"/>
    </visual-themes>
```

You need to define in the database an entry in VisualTheme Entity for each visual-theme id defined.

```
<entity-engine-xml>
    <VisualTheme visualThemeId="MY_THEME" visualThemeSetId="BACKOFFICE"
description="My theme - Example (based on flatgrey)"/>
</entity-engine-xml>
```

> ℹ a theme component can load one or more visual theme id. Usually only one is present.

The second important (but not mandatory) element is **implements**

```
   <extends location="component://common-theme/widget/Theme.xml"/>
```

This element indicates that your theme copies from the extend theme all information not present in its file definition.
If this element isn’t present in your theme, you will need to define all information present in common-theme to be sure that OFBiz misses nothing for a correct run. Otherwise some functionnalities can be broken…

> ℹ It’s highly recommended to extend the common-theme to be sure that your theme works correctly and to surcharge only what you need.

The four following blocks are optionnal if you define an extends theme

3.2. General properties

This block contains all properties that the screen engine can use to prepare the rendering and that the theme can implement

```
    <widget-properties><!--Transversal properties relative to ofbiz widget component--
>
        <default-view-size value="20"/>
        <autocompleter
                default-view-size="10"
                default-min-lenght="2"
                default-delay="300"
                display-return-field="true"/>
        <lookup
                position="topleft"
                width="640"
                height="500"/>
        <layered-modal
                width="800"
                height="600"/>
    </widget-properties>
```

3.3. Theme’s specific properties

This block contains all properties specific to this theme. In general these are some properties present on ftl template, that are initialized by the theme and can be surchaged by another theme through the extends elements.

```
        <property name="jgrowlPosition" value="center" type="String"/><!--possible
value: top-left, top-right, bottom-left, bottom-right, center-->
        <property name="jgrowlWidth" value="800" type="Integer"/>
        <property name="jgrowlHeight" value="" type="Integer"/>
        <property name="jgrowlSpeed" value="100" type="Integer"/>
```

3.4. Ftl macro library

This block defines for each technology implemented by the screen engine where it can find the macro library for each model (Screen, Form, Tree, Menu)

```
   <templates><!-- Freemarker template use by this theme to render widget model-->
        <template name="screen" type="html" content-type="UTF-8" encoding="none"
encoder="html" compress="false">
            <template-file widget="screen" location="component://common-
theme/template/macro/HtmlScreenMacroLibrary.ftl"/>
            <template-file widget="form" location="component://common-
theme/template/macro/HtmlFormMacroLibrary.ftl"/>
            <template-file widget="tree" location="component://common-
theme/template/macro/HtmlTreeMacroLibrary.ftl"/>
            <template-file widget="menu" location="component://common-
theme/template/macro/HtmlMenuMacroLibrary.ftl"/>
        </template>
        ...
   </templates>
```

If you want surcharge some macros, you can just create the desired macros and import the others from common-theme (at the top of file) like that : `<#include "component://common-theme/template/macro/HtmlFormMacroLibrary.ftl"/>`

3.5. Screen library

This block defines where OFBiz can find all official screens definitions in framework/common

Normally, you don’t need to change this file except if you need to define a default screen style that doesn’t exist OOTB. If you need to extend an existing one, you have to do it in the theme directory.

To define a new default screen style, you have to add it in this file, and point to the screen decorator in common-theme to define your default screen style as the default one.

```
     <common-screens><!--list all common screen and decorator global to each
application that each theme can be surcharge or not and use the screen present on
common theme-->
        <structural-decorator default-location="component://common-
theme/widget/CommonScreens.xml">
            <screen name="GlobalDecorator"/>
            ...
        </structural-decorator>
        <embed-decorator default-location="component://common-
theme/widget/CommonScreens.xml">
            <screen name="FindScreenDecorator"/>
        </embed-decorator>
        <general-screen default-location="component://common-
theme/widget/CommonScreens.xml">
            <screen name="geoChart"/>
            ...
        </general-screen>
    </common-screens>
```

Screens are separated in three types :

- structural-decorator : contains all decorators that organise the screens structures
- embed-decorator : decorator used only on sub screens
- general-screen : list all generic inter applications screens

4. The common-theme

This is the root theme that contains all information to ensure a good basic theme for OFBiz. Currently it keeps all old themes system for backward compatibility with ftl template managed by the entity **VisualThemeResource**

5. Create your own theme

As a theme is a component, you can create a new theme like a plugin.

After creating a component, you can add the two minimal information :

- Theme.xml file in **plugins/my-theme/widget/** with minimal information :

```
<theme name="my-theme"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="http://ofbiz.apache.org/dtds/widget-theme.xsd">
    <visual-themes>
        <visual-theme id="MY_THEME" display-name="My Theme"/>
    </visual-themes>
</theme>
```

- your data file to add your visual theme in **plugins/my-theme/data/**

```
<entity-engine-xml>
    <VisualTheme visualThemeId="MY_THEME" visualThemeSetId="BACKOFFICE"/>
</entity-engine-xml>
```

The presence of VisualTheme entity helps to indicate which theme is available in your instance, specially helpful for tenant installations.

To display your theme in OFBiz theme library, you can complete the information on each visual theme like

```
<theme name="my-theme"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="http://ofbiz.apache.org/dtds/widget-theme.xsd">
    <visual-themes>
        <visual-theme id="MY_THEME" display-name="My Theme">
            <description>My new funny theme under nice tecnno</description>
            <screenshot location="/mytheme/screenshot1.png"/>
            <screenshot location="/mytheme/screenshot2.png"/>
        </visual-theme>
    </visual-themes>
</theme>
```

display-name and **description** support the flexibleStringExpander syntax

5.1. extends common-theme

This is a first step to understand how the theme system works. With your new theme, you can try to surchage different elements.

To start, extends the common-theme :

```
<theme name="my-theme"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="http://ofbiz.apache.org/dtds/widget-theme.xsd">
    <visual-themes>
        <visual-theme id="MY_THEME" display-name="My Theme">
            <description>My new funny theme under nice tecnno</description>
            <screenshot location="/mytheme/screenshot1.png"/>
        </visual-theme>
    </visual-themes>
    <extends location="component://common-theme/widget/Theme.xml"/>
</theme>
```

Now your theme should be operational, but without particularity.

You can surcharge a ftl macro, to do this create your own ftl macro file in **plugins/my-theme/templates/macro/HtmlFormMacroLibrary.ftl** with

```
<#include "component://common-theme/template/macro/HtmlFormMacroLibrary.ftl"/>

<#macro renderDisplayField type imageLocation idName description title class alert
inPlaceEditorUrl="" inPlaceEditorParams="">
    <#if description?has_content>
    *###*${description?replace("\n", "<br />")}**<#t/>
    <#else>
        *#&nbsp;#*<#t/>
    </#if>
</#macro>
```

Now indicate to your theme that you want use this library

```
<theme name="my-theme"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="http://ofbiz.apache.org/dtds/widget-theme.xsd">
    <visual-themes>
        <visual-theme id="MY_THEME" display-name="My Theme">
            <description>My new funny theme under nice tecnno</description>
            <screenshot location="/mytheme/screenshot1.png"/>
        </visual-theme>
    </visual-themes>
    <extends location="component://common-theme/widget/Theme.xml"/>
    <templates>
        <template name="screen" type="html" content-type="UTF-8" encoding="none"
encoder="html" compress="false">
            <template-file widget="form" location="component://my-
theme/template/macro/HtmlFormMacroLibrary.ftl"/>
        </template>
    </templates>
</theme>
```

and check the result when you select your theme. The result isn’t really interesting but it’s to understand how it works.

5.2. create from scratch

TODO…

6. Backware compatibility with OFBiz 16.11 and above

6.1. How themes worked before

Before the theme management by model definition, all configurations have been present in the database through entity **VisualTheme** and **VisualThemeRessource**.
These ressources were loaded in a **layoutProperties** variable and used directly by decorator screens and ftl templates.

6.2. Now with the common-theme

All this logic is still present in the common-theme template to keep backward compatibility, but the VisualThemeRessource is now useless and properties have been migrated to the Theme definition in the part **theme-properties**

6.2.1. Example with BlueLight

The blue light theme has been these properties in VisualThemeRessource :

```
    <VisualTheme visualThemeId="BLUELIGHT" visualThemeSetId="BACKOFFICE"
description="BlueLight Theme: breadcrumbs, drop-down menus and rounded corners"/>
    <VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_NAME"
resourceValue="BLUELIGHT" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_IMAGE_URL" resourceValue="/images/ofbiz_logo.png"
sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_SHORTCUT_ICON" resourceValue="/images/ofbiz.ico"
sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_SCREENSHOT"
resourceValue="/bluelight/screenshot.jpg" sequenceId="01"/>

    <!-- CSS references -->
    <VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_STYLESHEET"
resourceValue="/bluelight/style.css" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HELPSTYLESHEET" resourceValue="/bluelight/help.css"
sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_DOCBOOKSTYLESHEET"
resourceValue="/bluelight/webapp/bluelight/docbook.css" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_STYLESHEET"
resourceValue="/common/js/jquery/plugins/asmselect/jquery.asmselect-1.0.4a-beta.css"
sequenceId="02"/>

    <!-- Javascript references -->
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/jquery-
1.11.0.min.js" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/jquery-
migrate-1.2.1.js" sequenceId="02"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/ui/js/jquery-
ui-1.10.3.min.js" sequenceId="03"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/jquery/plugins/asmselect/jquery.asmselect-1.0.4a-beta.js"
sequenceId="05"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/jquery/plugins/datetimepicker/jquery-ui-timepicker-
addon.min-1.4.3.js" sequenceId="07"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/jquery/plugins/fjTimer/jquerytimer-min.js" sequenceId="09"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/jquery/plugins/mask/jquery.mask-1.14.13.min.js"
sequenceId="10"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/jquery/plugins/jeditable/jquery.jeditable-1.7.3.js"
sequenceId="11"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/jquery/plugins/validate/jquery.validate.min.js"
sequenceId="12"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/plugins/OpenLayers-
2.13.1.js" sequenceId="13"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/OfbizUtil.js"
sequenceId="15"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/fieldlookup.js"
sequenceId="16"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/plugins/date/date.format-1.2.3-min.js" sequenceId="17"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/plugins/date/date.timezone-min.js" sequenceId="18"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/util/miscAjaxFunctions.js" sequenceId="19"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
```

```
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/util/selectMultipleRelatedValues.js" sequenceId="20"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/util.js"
sequenceId="21"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT"
resourceValue="/common/js/plugins/date/FromThruDateCheck.js" sequenceId="22"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/bluelight/dropdown.js"
sequenceId="30"/>

    <!-- ftl references -->
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_HDR_TMPLT_LOC"
resourceValue="component://bluelight/template/Header.ftl" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_FTR_TMPLT_LOC"
resourceValue="component://bluelight/template/Footer.ftl" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_NAV_OPEN_TMPLT"
resourceValue="component://bluelight/template/AppBarOpen.ftl" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_NAV_CLOSE_TMPLT"
resourceValue="component://bluelight/template/AppBarClose.ftl" sequenceId="01"/>
    <VisualThemeResource visualThemeId="BLUELIGHT"
resourceTypeEnumId="VT_MSG_TMPLT_LOC"
resourceValue="component://bluelight/template/Messages.ftl" sequenceId="01"/>
```

Now it’s just

```
    <VisualTheme visualThemeId="BLUELIGHT" visualThemeSetId="BACKOFFICE"/>
```

And on theme definition

```
    <theme-properties>
        <!--javascript lib-->
        <property name="VT_HDR_JAVASCRIPT['add']" value="/bluelight/dropdown.js"
sequenceId="30"/>
        <!--Css style-->
        <property name="VT_STYLESHEET['add']" value="/bluelight/style.css"/>
        <property name="VT_HELPSTYLESHEET['add']" value="/bluelight/help.css"/>
        <property name="VT_DOCBOOKSTYLESHEET['add']"
value="/bluelight/webapp/bluelight/docbook.css"/>
        <!--template location-->
        <property name="VT_HDR_TMPLT_LOC"
value="component://bluelight/template/Header.ftl"/>
        <property name="VT_FTR_TMPLT_LOC"
value="component://bluelight/template/Footer.ftl"/>
        <property name="VT_NAV_OPEN_TMPLT"
value="component://bluelight/template/AppBarOpen.ftl"/>
        <property name="VT_NAV_CLOSE_TMPLT"
value="component://bluelight/template/AppBarClose.ftl"/>
        <property name="VT_MSG_TMPLT_LOC"
value="component://bluelight/template/Messages.ftl"/>
    </theme-properties>
```

Values with **/images/…** have been moved to the common-theme that bluelight extends, the theme definition keeps only what the theme adds to the extended theme.

> property name supports the FlexibleMapAccessor syntax, so you can continue to populate a list (VT_STYLESHEET['add']), reset a list (VT_STYLESHEET[]) or add an element on the top list (VT_STYLESHEET[+0]) because some time the order libraries loading is important

6.2.2. Migrate you own theme

Easily, create you Theme.xml and move your VisualThemeResource in **theme-properties** like in the BlueLight example above.
Maybe you will need to update your template because the modelTheme return ressources not always as list. So :
`<property name="VT_HDR_TMPLT_LOC" value="component://bluelight/template/Header.ftl"/>` → return a String with `component://bluelight/template/Header.ftl`
`<property name="VT_STYLESHEET['add'] value="…"` → return a List<String>