eXo Content Reference Guide

eXo Content

Benjamin Paillereau (eXo Platform)

Chapter 1. Preface

1.1. Getting started with eXo WCM

1.2. Packaging

1.3. Portlets

1.3.1. Content Detail

1.3.1.1. Available preferences

Preference	Type	Default Value	Description
repository	`String`	repository	The current repository name. always is "repository"
workspace	`String`	collaboration	The workspace where the content is stored
nodeIdentifier	`String`	N/A	The UUID or the path of content that you want to show
ShowTitle	`Boolean`	true	Show the content title on top of the portlet
ShowDate	`Boolean`	false	Show the content date on top of the portlet
ShowOptionBar	`Boolean`	false	Show a bar with some actions (Print, Back)
ContextEnable	`Boolean`	false	Determine if the portlet will use the parameter on URL as the path to content to display
ParameterName	`String`	content-id	Determine what parameter will be used to get the content's path
ShowVote	`Boolean`	false	Show the result of voting for the displayed content.
ShowComments	`Boolean`	false	Show the existing comments of this content

Preference	Type	Default Value	Description
			(if any)

1.3.1.2. Sample configuration

```
<portlet-preferences>
  <preference>
    <name>repository</name>
    <value>repository</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>workspace</name>
    <value>collaboration</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>nodeIdentifier</name>
    <value>/myfolder/mycontent</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>ShowTitle</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>ShowDate</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>ShowOptionBar</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>ShowPrintAction</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>isQuickCreate</name>
    <value>false</value>
    <read-only>true</read-only>
  </preference>
  <preference>
    <name>ContextEnable</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>ParameterName</name>
    <value>content-id</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

Note: In 2.1.x, there is some preference is no longer used: for example: ShowPrintAction preference,..

1.3.2. Content List

1.3.2.1. Available preferences

Preference	Type	Default Value	Description
mode	`String`	AutoViewerMode	The mode that the portlet uses to display content: all contents in a specified folder or all contents specified in the portlet.
folderPath	`String`		The path to the folder whose contents are displayed by this portlet.
orderBy	`String`	publication:liveDate	The property by which all the contents in portlet are sorted.
orderType	`String`	DESC	The type of the content sort method: ascending or descending
header	`String`		Header of the portlet. It is displayed at the top of the portlet.
automaticDetection	`Boolean`	true	This value indicates whether the header of the portlet is chosen to be the title of the folder given in the **folderPath** parameter (true value) or the value given in the **header** paramter above
formViewTemplatePath	`String`	/exo:ecm/views/templates/ContentList Viewer/list-by-folder/UIContentListPresentationDefault.gtmpl	Path to the template used to display the contents in this portlet.
paginatorTemplatePath	`String`	/exo:ecm/views/templates/ContentList Viewer/paginators/UIPaginatorDefault.gtmpl	Path to the paginator used to display the contents in this portlet.
itemsPerPage	`Integer`	10	Number of contents displayed in every "page" of the portlet.
showThumbnailsView	`Boolean`	true	This value indicates whether the content image in this portlet is shown or not.
showTitle	`Boolean`	true	This value indicates whether the content title in this portlet is shown or not.
showHeader	`Boolean`	true	This value indicates whether the content

Preference	Type	Default Value	Description
			header in this portlet is shown or not.
showRefreshButton	`Boolean`	false	This value indicates whether the **refresh** button is shown in this portlet or not.
showDateCreated	`Boolean`	true	This value indicates whether the content created date in this portlet is shown or not.
showReadmore	`Boolean`	true	This value indicates whether the Readmore" button is shown in every content of the portlet (After clicking this button, user can read the whole text of the content).
showSummary	`Boolean`	true	This value indicates whether the content summary in this portlet is shown or not.
showLink	`Boolean`	true	If this value is **true**, the header of every content is also the link to view this content fully. If the value is **false**, the header is considered as a simple text
showRssLink	`Boolean`	true	Show the rss link of this portlet.
basePath	`String`	detail	Show the page in which the full content is displayed when user clicks to the **Read more** button.
contextualFolder	`String`	contextualDisable	Enable/ disable the contextual mode of the portlet. If enabled, the portlet can take the folder path indicated in the URL to display contents.
showScvWith	`String`	content-id	The name of the parameter showing the path of content in url to

Preference	Type	Default Value	Description
			display when click on "Read more" button of this content.
showClvBy	`String`	folder-id	The name of the parameter showing the path of the folder in url to display its contents.

1.3.2.2. Sample Configuration

```
<portlet-preferences>
  <preference>
    <name>mode</name>
    <value>AutoViewerMode</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>folderPath</name>
    <value/>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>orderBy</name>
    <value>publication:liveDate</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>orderType</name>
    <value>DESC</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>header</name>
    <value/>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>automaticDetection</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>formViewTemplatePath</name>
    <value>/exo:ecm/views/templates/Content List Viewer/list-by-folder/UIContentListPresentationDefault.gtmpl</v
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>paginatorTemplatePath</name>
    <value>/exo:ecm/views/templates/Content List Viewer/paginators/UIPaginatorDefault.gtmpl</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>itemsPerPage</name>
    <value>10</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showThumbnailsView</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showTitle</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showHeader</name>
```

```
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showRefreshButton</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showDateCreated</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showReadmore</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showSummary</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showLink</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showRssLink</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>basePath</name>
    <value>detail</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>contextualFolder</name>
    <value>contextualDisable</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showScvWith</name>
    <value>content-id</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showClvBy</name>
    <value>folder-id</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

1.3.3. Search

1.3.3.1. Available preferences

Preference	Type	Default value	Description
repository	`String`	repository	The current repository name. always is "repository"
workspace	`String`	collaboration	The workspace where the content is stored
searchFormTemplatePath	`String`	/exo:ecm/views/templates/wcm	Path to the search form

Preference	Type	Default value	Description
		Advance Search/search-form/UIDefaultSearchForm.gtmpl	template
searchResultTemplatePath	`String`	/exo:ecm/views/templates/WCM Advance Search/search-result/UIDefaultSearchResult.gtmpl	path to the search result template
searchPaginatorTemplatePath	`String`	/exo:ecm/views/templates/WCM Advance Search/search-paginator/UIDefaultSearchPaginator.gtmpl	path to the search paginator template
searchPageLayoutTemplatePath	`String`	/exo:ecm/views/templates/WCM Advance Search/search-page-layout/UISearchPageLayoutDefault.gtmpl	path to the search page template
itemsPerPage	`Integer`	5	the number of items for each page
showQuickEditButton	`Boolean`	true	show or hide the quick edit icon (!)
basePath	`String`	parameterizedviewer	the page which used to display the search result

1.3.3.2. Sample configuration

```
<portlet-preferences>
  <preference>
    <name>repository</name>
    <value>repository</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>workspace</name>
    <value>collaboration</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>searchFormTemplatePath</name>
    <value>/exo:ecm/views/templates/WCM Advance Search/search-form/UIDefaultSearchForm.gtmpl</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>searchResultTemplatePath</name>
    <value>/exo:ecm/views/templates/WCM Advance Search/search-result/UIDefaultSearchResult.gtmpl</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>searchPaginatorTemplatePath</name>
    <value>/exo:ecm/views/templates/WCM Advance Search/search-paginator/UIDefaultSearchPaginator.gtmpl</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>searchPageLayoutTemplatePath</name>
    <value>/exo:ecm/views/templates/WCM Advance Search/search-page-layout/UISearchPageLayoutDefault.gtmpl</value
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>itemsPerPage</name>
    <value>5</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showQuickEditButton</name>
    <value>true</value>
```

```
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>basePath</name>
    <value>parameterizedviewer</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

1.3.4. Explorer

1.3.4.1. Available preferences

Preference	Type	Default value	Description
repository	`String`	repository	The repository name which will be used in an instance of Site Explorer
workspace	`String`	N/A	Not in used. The workspace name was included in the Drive(!)
path	`String`	N/A	The path of node. This preference will be used when you choose the usecase is Parameterize
drive	`String`	N/A	Not in used. Replaced by driveName preference(!)
views	`String`	N/A	Not in used. The views will be displayed based on the Drive which user has access permission(!)
allowCreateFolders	`String`	N/A	Allow to create a folder by type. When you don't specify the value then the default value will be nt:unstructured, nt:folder
categoryMandatoryWhenFileUpload	`Boolean`	false	Force an user to add a category when uploading or creating a document
uploadFileSizeLimitMB	`Float`	150	The maximum of file size when upload to the system (MB)
usecase	`String`	selection	The behavior to access Site Explorer. By default the "selection" option is configured. Besides "selection", there are 4 other ways to configure

Preference	Type	Default value	Description
			the file explorer: **Jailed**,**Personal**,**Social**,**Parameterize**
driveName	`String`	Private	The name of a drive that an user wants to access
trashHomeNodePath	`String`	/Trash	The location to store the deleted nodes
trashRepository	`String`	repository	The repository name where store the deleted nodes
trashWorkspace	`String`	collaboration	The workspace name where store the deleted nodes
editInNewWindow	`Boolean`	false	Allow to edit document with or without window popup. By default the value is false
showTopBar	`Boolean`	true	Allow to show the Top bar or not. By default the value is true
showActionBar	`Boolean`	true	Allow to show action bar or not. By default the value is true
showSideBar	`Boolean`	true	Allow to show the side bar or not. By default the value is true
showFilterBar	`Boolean`	true	Allow to show the Filter bar or not. By default the value is true

1.3.4.2. Example of configuration

```
<portlet-preferences>
  <preference>
    <name>repository</name>
    <value>repository</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>workspace</name>
    <value/>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>path</name>
    <value/>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>drive</name>
    <value/>
    <read-only>false</read-only>
```

```
  </preference>
  <preference>
    <name>views</name>
    <value/>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>allowCreateFolders</name>
    <value/>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>categoryMandatoryWhenFileUpload</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>uploadFileSizeLimitMB</name>
    <value>150</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>usecase</name>
    <value>selection</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>driveName</name>
    <value>Private</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>trashHomeNodePath</name>
    <value>/Trash</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>trashRepository</name>
    <value>repository</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>trashWorkspace</name>
    <value>collaboration</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>editInNewWindow</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showTopBar</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showActionBar</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showSideBar</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>showFilterBar</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

1.3.5. Administration

1.3.5.1. Available preferences

Preference	Type	Default	Description
repository	`String`	repository	The current repository name. Default value is "repository"

1.3.5.2. Sample Configuration

```
<portlet-preferences>
  <preference>
    <name>repository</name>
    <value>repository</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

1.3.6. Fast Content Creator

1.3.6.1. Available preferences

Preference	Type	Default Value	Description
mode	`String`	basic	The default mode for fast content creator portlet
repository	`String`	repository	The current repository name. always is "repository"
workspace	`String`	collaboration	The workspace where the content is stored
path	`String`	/Groups/platform/users/Documents	The destination path where the content is stored
type	`String`	exo:article	The node type of document which will be show on the dialog form
saveButton	`String`	Save	The custom button Save
saveMessage	`String`	This node has been saved successfully	The custom message when click Save button
isRedirect	`Boolean`	false	Redirect to other page or not
redirectPath	`String`	http://www.google.com.vn	The page will redirect to
isActionNeeded	`Boolean`	true	Is an action needed to save the configuration

1.3.6.2. Example Configuration

```
<portlet-preferences>
  <preference>
    <name>mode</name>
    <value>basic</value>
    <read-only>true</read-only>
  </preference>
  <preference>
    <name>repository</name>
    <value>repository</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>workspace</name>
    <value>collaboration</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>path</name>
    <value>/Groups/platform/users/Documents</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>type</name>
    <value>exo:article</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>saveButton</name>
    <value>Save</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>saveMessage</name>
    <value>This node has been saved successfully</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>isRedirect</name>
    <value>false</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>redirectPath</name>
    <value>http://www.google.com.vn</value>
    <read-only>false</read-only>
  </preference>
  <preference>
    <name>isActionNeeded</name>
    <value>true</value>
    <read-only>true</read-only>
  </preference>
</portlet-preferences>
```

Chapter 2. Configuration

2.1. Drives

A drive is like a shortcut in the content repository, a quick access to some places for users. You can restrict the visibility of this drive to a group/user and apply a specific view depending on the content you have in this area.

For shorter, a drive is the combination of :

- Path : the root folder of the drive.
- View : how we can see the contents (by list, thumbnails, coverflow, etc)
- Role : visible to everybody, a group or a single user
- Options : can we see hidden nodes ? can we create folders in this drive ?

2.1.1. Fields

		org.exoplatform.services.cms.drives.DriveDa
name	`String`	Name of the drive, must be unique inside WCM
repository	`String`	Content Repository where to find the root path
workspace	`String`	Workspace in the Content Repository
homePath	`String`	root path in the Content Repository. `_${userId}_` can be used to use the userId at runtime in the path.
permissions	`String`	Visibility of the drive based on eXo Rights
	example	`*:/platform/users`
icon	`String`	url to the icon
views	`String`	List of views you want to use
	example	`simple-view,admin-view`
viewPreferences	`Boolean`	"User Preference" icon will be visible if true
viewNonDocument	`Boolean`	Non document types will be visible

		org.exoplatform.services.cms.drives.DriveDa
		in the user view if true
viewSideBar	`Boolean`	Show/Hide the left bar (with navigation and filters)
showHiddenNode	`Boolean`	Hidden nodes will be visible if true
allowCreateFolders	`Boolean`	List of nodetype we can create as folders
	example	`nt:folder,nt:unstructured`
allowNodeTypesOnTree	`String`	Allows you to filter nodetypes in the navigation tree. Default value is "*" to show all content types
	example	`*`
		`exo:taxonomy`

We use the following structure for drives configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.cms.drives.ManageDriveService</target-component>
  <component-plugin>
    <name>manage.drive.plugin</name>
    <set-method>setManageDrivePlugin</set-method>
    <type>org.exoplatform.services.cms.drives.impl.ManageDrivePlugin</type>
    <description>Nothing</description>
    <init-params>
      <object-param>There are initializing attributes of org.exoplatform.services.cms.drives.DriveData object</o
    </init-params>
  </component-plugin>
</external-component-plugins>
```

The file that contains the structure above will be configured in configuration.xml file as the following:

```
<import>war:/conf/wcm-extension/dms/drives-configuration.xml</import>
```

2.1.2. Example of configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.cms.drives.ManageDriveService</target-component>
  <component-plugin>
    <name>manage.drive.plugin</name>
    <set-method>setManageDrivePlugin</set-method>
    <type>org.exoplatform.services.cms.drives.impl.ManageDrivePlugin</type>
    <description>Nothing</description>
    <init-params>
      <object-param>
        <name>Managed Sites</name>
        <description>Managed Sites</description>
        <object type="org.exoplatform.services.cms.drives.DriveData">
          <field name="name">
            <string>Managed Sites</string>
          </field>
          <field name="repository">
            <string>repository</string>
          </field>
          <field name="workspace">
            <string>collaboration</string>
          </field>
```

```
          <field name="permissions">
            <string>*:/platform/administrators</string>
          </field>
          <field name="homePath">
            <string>/sites content/live</string>
          </field>
          <field name="icon">
            <string/>
          </field>
          <field name="views">
            <string>wcm-view</string>
          </field>
          <field name="viewPreferences">
            <boolean>false</boolean>
          </field>
          <field name="viewNonDocument">
            <boolean>true</boolean>
          </field>
          <field name="viewSideBar">
            <boolean>true</boolean>
          </field>
          <field name="showHiddenNode">
            <boolean>false</boolean>
          </field>
          <field name="allowCreateFolders">
            <string>nt:folder,nt:unstructured</string>
          </field>
          <field name="allowNodeTypesOnTree">
            <string>*</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>Public</name>
        <description>Public drive</description>
        <object type="org.exoplatform.services.cms.drives.DriveData">
          <field name="name">
            <string>Public</string>
          </field>
          <field name="repository">
            <string>repository</string>
          </field>
          <field name="workspace">
            <string>collaboration</string>
          </field>
          <field name="permissions">
            <string>*:/platform/users</string>
          </field>
          <field name="homePath">
            <string>/Users/${userId}/Public</string>
          </field>
          <field name="icon">
            <string/>
          </field>
          <field name="views">
            <string>simple-view, admin-view</string>
          </field>
          <field name="viewPreferences">
            <boolean>false</boolean>
          </field>
          <field name="viewNonDocument">
            <boolean>false</boolean>
          </field>
          <field name="viewSideBar">
            <boolean>true</boolean>
          </field>
          <field name="showHiddenNode">
            <boolean>false</boolean>
          </field>
          <field name="allowCreateFolders">
            <string>nt:folder,nt:unstructured</string>
          </field>
          <field name="allowNodeTypesOnTree">
            <string>*</string>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
```

```
</external-component-plugins>
```

2.2. Publication

2.3. Deployment

When a site is created, normally the end-user want to see something in the page instead of a blank page, so we need this service for deployment some "default" contents like Banner, Footer, Navigation, Breadcrumb, ... There are too main case to use

- The site is created only one time when the database is clean.
- The site is created at runtime, when user use the core feature of GateIn portal.

2.3.1. Fields

`init-params`

Element	Type	Default value	Description
object-param	`Object`	(see below)	The parameter which is an Object

`object-param`

Element	Type	Default value	object-param
name	`String`		The name of this object parameter
description	`String`		The description of this object parameter
object	`Class`	(see below)	The object of this object parameter

`object`

Attribute	Type	Default value	Description
type	`String`	org.exoplatform.services.deployment.DeploymentDescriptor (*)	The type of the object

`org.exoplatform.services.deployment.DeploymentDescriptor`

Name	Type	Default value	Description
target	`Object`	org.exoplatform.services.deployment.DeploymentDescriptor$Target (*)	The target node (which will contains the imported node)
sourcePath	`String`		The absolute path of the XML file

`org.exoplatform.services.deployment.DeploymentDescriptor$Target`

Field	Type	Default value	Description
repository	`String`		The repository of the target node
workspace	`String`		The workspace of the target node
nodePath	`String`		The path of the target node

2.3.2. Example of configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.deployment.WCMContentInitializerService</target-component>
  <component-plugin>
    <name>Content Initializer Service</name>
    <set-method>addPlugin</set-method>
    <type>org.exoplatform.services.deployment.plugins.XMLDeploymentPlugin</type>
    <description>XML Deployment Plugin</description>
    <init-params>
      <object-param>
        <name>ACME Logo data</name>
        <description>Deployment Descriptor</description>
        <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
          <field name="target">
            <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
              <field name="repository">
                <string>repository</string>
              </field>
              <field name="workspace">
                <string>collaboration</string>
              </field>
              <field name="nodePath">
                <string>/sites content/live/acme/web contents/site artifacts</string>
              </field>
            </object>
          </field>
          <field name="sourcePath">
            <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme/Logo.xml</string>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

2.4. Taxonomies

Taxonomies are used to sort documents in order to ease searches when browsing documents online. The main idea behind that concept is to provide a multi dimensional set of paths to find a document. The best example is when you browse Yahoo news. In many cases, you can get your content by using different category paths. Therefore, after creating a document somewhere in the repository, it is possible to categorize it by adding several taxonomy references. By browsing the taxonomy tree, it will be possible to find the referencing article and display them as if they were children of the taxonomy nodes. As you can imagine, taxonomies are stored in the JCR itself and we are using the JCR Reference functionality to provide that advanced ECM feature.

Managing the tree of taxonomies is very simple, you can copy/cut nodes and paste them. Of course you can add and remove taxonomies from the tree. Once a taxonomy has been added, any user who has access to the "Manage Categories" icon from his/her view can then browse the taxonomy tree and refer one of its nodes from the created documents.

2.4.1. Fields

		org.exoplatform.services.cms.taxonomy.impl.
permissions	`java.util.ArrayList<TaxonomyTreeDefaultUserPermission>`	List of default permission to access the Taxonomy Tree

		org.exoplatform.services.cms.taxonomy.impl.
identity	`String`	user name or user group.
read	`Boolean`	permission to read taxonomy tree.
addNode	`Boolean`	permission to add a node in taxonomy tree.
setProperty	`Boolean`	permission to set properties for node in taxonomy tree .
remove	`Boolean`	permission to remove node in taxonomy tree.

		org.exoplatform.services.cms.taxonomy.impl.
autoCreateInNewRepository	`Boolean`	Enable/ Disable the creation of the taxonomies in new created repository.
repository	`String`	name of the repository where taxonomies are created.
workspace	`String`	name of the workspace where taxonomies are created.
treeName	`String`	name of the taxonomy tree to be created.
permission.configuration	`org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig`	access permission for the whole taxonomy tree
predefined.actions	`org.exoplatform.services.cms.taxonomy.impl.ActionsConfig`	predefined actions for the

		org.exoplatform.services.cms.taxonomy.impl.
		taxonomy tree root node.
taxonomy.configuration	`org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig`	access permissions for each taxonomy in tree.

		org.exoplatform.services.cms.taxonomy.impl.
taxonomies	`java.util.ArrayList<TaxonomyConfig.Taxonomy>`	list of taxonomy to be configured permission.

		org.exoplatform.services.cms.taxonomy.impl.
name	`String`	name of the taxonomy.
path	`String`	path to the taxonomy in taxonomy tree.
permissions	`java.util.ArrayList<TaxonomyTreeListener.PermissionConfig.Permission>`	list of permission for user/group to access the taxonomy

		org.exoplatform.services.cms.actions.impl.Ac
actions	`java.util.ArrayList<ActionConfig.TaxonomyAction>`	list of actions which are created for the taxonomy tree root node.

		org.exoplatform.services.cms.actions.impl.Ac
type	`String`	type of the action.
name	`String`	name of the action.
description	`String`	description of the action.
homePath	`String`	location of node where the action node is stored.
targetWspace	`String`	when a new node is created in the node whose path is defined in **homePath** field, it will be moved to a new location automatically. The target workspace is specified in this field.
targetPath	`String`	when a new node is created in the node whose path is defined in **homePath** field, it will be moved to a new location automatically. The target path is specified in this field.
lifecyclePhase	`java.util.ArrayList<String>`	the life cycle phases, in which the

		org.exoplatform.services.cms.actions.impl.Ac
		action is activated.
roles	`String`	the users or groups only with whom the action is activated.
mixins	`java.util.ArrayList<ActionConfig$Mixin>`	list of node types that are affected by this action.

		org.exoplatform.services.cms.actions.impl.Ac
name	`String`	mixin node type name that will be added, whose responsibility is to store the affected node types.
properties	`String`	name of the properties that stores all the node types affected by the action.

2.4.2. Sample Configuration

```
<component>
  <key>org.exoplatform.services.cms.taxonomy.TaxonomyService</key>
  <type>org.exoplatform.services.cms.taxonomy.impl.TaxonomyServiceImpl</type>
  <init-params>
    <object-param>
      <name>defaultPermission.configuration</name>
      <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyTreeDefaultUserPermission">
        <field name="permissions">
          <collection type="java.util.ArrayList">
            <value>
              <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyTreeDefaultUserPermission$Permiss
                <field name="identity">
                  <string>*:/platform/administrators</string>
                </field>
                <field name="read">
                  <string>true</string>
                </field>
                <field name="addNode">
                  <string>true</string>
                </field>
                <field name="setProperty">
                  <string>true</string>
                </field>
                <field name="remove">
                  <string>true</string>
                </field>
              </object>
            </value>
          </collection>
        </field>
      </object>
    </object-param>
  </init-params>
  <external-component-plugins>
    <target-component>org.exoplatform.services.cms.taxonomy.TaxonomyService</target-component>
    <component-plugin>
      <name>predefinedTaxonomyPlugin</name>
      <set-method>addTaxonomyPlugin</set-method>
      <type>org.exoplatform.services.cms.taxonomy.impl.TaxonomyPlugin</type>
      <init-params>
        <value-param>
          <name>autoCreateInNewRepository</name>
          <value>true</value>
        </value-param>
```

```
        <value-param>
          <name>repository</name>
          <value>repository</value>
        </value-param>
        <value-param>
          <name>workspace</name>
          <value>dms-system</value>
        </value-param>
        <value-param>
          <name>treeName</name>
          <value>System</value>
        </value-param>
        <object-param>
          <name>permission.configuration</name>
          <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig">
            <field name="taxonomies">
              <collection type="java.util.ArrayList">
                <value>
                  <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig$Taxonomy">
                    <field name="permissions">
                      <collection type="java.util.ArrayList">
                        <value>
                          <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig$Permission">
                            <field name="identity">
                              <string>*:/platform/users</string>
                            </field>
                            <field name="read">
                              <string>true</string>
                            </field>
                            <field name="addNode">
                              <string>true</string>
                            </field>
                            <field name="setProperty">
                              <string>true</string>
                            </field>
                            <field name="remove">
                              <string>false</string>
                            </field>
                          </object>
                        </value>
                      </collection>
                    </field>
                  </object>
                </value>
              </collection>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>predefined.actions</name>
          <description>description</description>
          <object type="org.exoplatform.services.cms.actions.impl.ActionConfig">
            <field name="actions">
              <collection type="java.util.ArrayList">
                <value>
                  <object type="org.exoplatform.services.cms.actions.impl.ActionConfig$TaxonomyAction">
                    <field name="type">
                      <string>exo:taxonomyAction</string>
                    </field>
                    <field name="name">
                      <string>taxonomyAction</string>
                    </field>
                    <field name="description">
                      <string/>
                    </field>
                    <field name="homePath">
                      <string>dms-system:/exo:ecm/exo:taxonomyTrees/storage/System</string>
                    </field>
                    <field name="targetWspace">
                      <string>collaboration</string>
                    </field>
                    <field name="targetPath">
                      <string>/Documents</string>
                    </field>
                    <field name="lifecyclePhase">
                      <collection type="java.util.ArrayList">
                        <value>
                          <string>node_added</string>
                        </value>
```

```
                    </collection>
                  </field>
                  <field name="roles">
                    <string>*:/platform/administrators</string>
                  </field>
                  <field name="mixins">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.cms.actions.impl.ActionConfig$Mixin">
                          <field name="name">
                            <string>mix:affectedNodeTypes</string>
                          </field>
                          <field name="properties">
                            <string>exo:affectedNodeTypeNames=exo:article,exo:podcast,exo:sample,kfx:document,
                          </field>
                        </object>
                      </value>
                    </collection>
                  </field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>taxonomy.configuration</name>
        <description>configuration predefined taxonomies to inject in jcr</description>
        <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig">
          <field name="taxonomies">
            <collection type="java.util.ArrayList">
              <!-- cms taxonomy -->
              <value>
                <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig$Taxonomy">
                  <field name="name">
                    <string>cmsTaxonomy</string>
                  </field>
                  <field name="path">
                    <string>/cms</string>
                  </field>
                  <field name="permissions">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig$Permission">
                          <field name="identity">
                            <string>*:/platform/users</string>
                          </field>
                          <field name="read">
                            <string>true</string>
                          </field>
                          <field name="addNode">
                            <string>true</string>
                          </field>
                          <field name="setProperty">
                            <string>true</string>
                          </field>
                          <field name="remove">
                            <string>false</string>
                          </field>
                        </object>
                      </value>
                    </collection>
                  </field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
</component>
```

2.5. Nodetypes

Every node has a type. A node's type The names, types and other attributes of its child items. Node types can be used to define complex storage objects consisting of multiple sub nodes and properties, possibly many layers deep.

2.5.1. Fields

		org.exoplatform.services.jcr.core.nodetype.E
name	`String`	Name of the Node type, this field is required
isMixin	`boolean`	mixin type
hasOrderableChildNodes	`boolean`	Orderable childnodes
primaryItemName	`String`	Primary item name
suppertype	`String`	Name of supper type
propertyDefinition	`Object`	Define properties of node type

2.5.2. Examples Nodetype configuration

The node definition should be placed in a special XML file (see DTD below) and declared in the service's configuration file to eXo component plugin mechanism, described as follows:

```
       <external-component-plugins>
         <target-component>org.exoplatform.services.jcr.RepositoryService</target-component>
               <component-plugin>
           <name>add.namespaces</name>
           <set-method>addPlugin</set-method>
           <type>org.exoplatform.services.jcr.impl.AddNamespacesPlugin</type>
           <init-params>
             <properties-param>
               <name>namespaces</name>
               <property name="publication" value="http://www.exoplatform.com/jcr/publication/1.1/"/>
             </properties-param>
           </init-params>
         </component-plugin>
         <component-plugin>
           <name>add.nodeType</name>
           <set-method>addPlugin</set-method>
           <type>org.exoplatform.services.jcr.impl.AddNodeTypePlugin</type>
                       <priority>99</priority>
           <init-params>
             <values-param>
               <name>autoCreatedInNewRepository</name>
               <description>Node types configuration file</description>
               <value>jar:/conf/nodetypes-publication-config.xml</value>
             </values-param>
           </init-params>
         </component-plugin>
       </external-component-plugins>
```

This is a Nodetype definition file format:

```
<nodeTypes xmlns:nt="http://www.jcp.org/jcr/nt/1.0" xmlns:mix="http://www.jcp.org/jcr/mix/1.0"
  xmlns:jcr="http://www.jcp.org/jcr/1.0">
  <nodeType name="publication:publication" isMixin="true" hasOrderableChildNodes="false" primaryItemName="">
    <propertyDefinitions>
            <propertyDefinition name="publication:lifecycleName" requiredType="String" autoCreated="false" manda
```

```
        onParentVersion="COPY" protected="false" multiple="false">
        <valueConstraints/>
      </propertyDefinition>
      <propertyDefinition name="publication:currentState" requiredType="String" autoCreated="false" mandatory="t
        onParentVersion="COPY" protected="false" multiple="false">
        <valueConstraints/>
      </propertyDefinition>
      <propertyDefinition name="publication:history" requiredType="String" autoCreated="false" mandatory="true"
        onParentVersion="COPY" protected="false" multiple="true">
        <valueConstraints/>
      </propertyDefinition>
    </propertyDefinitions>
  </nodeType>
</nodeTypes>
```

2.6. Views

A view can include many object parameters. Parameters are used to create default views, templates and actions of Manage view service. View allow administrators to customize many sort of views that can impact to users in exploring workspace. Each object-param have a type that is a class representing all properties of a view.

2.6.1. Fields

org.exoplatform.services.cms.views.ViewConfig

Name	Type	Description
name	`String`	view name, must be unique inside WCM
permissions	`String`	Visibility of the view based on eXo Rights
	example	`*:/platform/administrators`
template	`String`	Specify path to the template location
tabList	`java.util.ArrayList`	include a set of view names.

org.exoplatform.services.cms.views.ViewConfig$Tab

Name	Type	Description
tabName	`String`	tab name that is not uniquely
button	`String`	These have to specify a set of view component names
	example	viewNodeType; viewPermissions; viewProperties; showJCRStructure

org.exoplatform.services.cms.views.TemplateConfig

Name	Type	Description
type	`String`	Specify to a name that is truly a class representing all properties of a view.
name	`String`	These have to specify a set of view component names
warPath	`String`	Specify to a template location for viewing
	example	/ecm-explorer/SystemView.gtmpl

2.6.2. Example of configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.cms.views.ManageViewService</target-component>
  <component-plugin>
    <name>manage.view.plugin</name>
    <set-method>setManageViewPlugin</set-method>
    <type>org.exoplatform.services.cms.views.impl.ManageViewPlugin</type>
    <description>this plugin manage user view</description>
    <init-params>
      <value-param>
        <name>autoCreateInNewRepository</name>
        <value>true</value>
      </value-param>
      <value-param>
        <name>predefinedViewsLocation</name>
        <value>war:/conf/dms-extension/dms/artifacts</value>
      </value-param>
      <value-param>
        <name>repository</name>
        <value>repository</value>
      </value-param>
      <object-param>
        <name>System-View</name>
        <description>View configuration for System workspace</description>
        <object type="org.exoplatform.services.cms.views.ViewConfig">
          <field name="name">
            <string>system-view</string>
          </field>
          <field name="permissions">
            <string>*:/platform/administrators</string>
          </field>
          <field name="template">
            <string>/exo:ecm/views/templates/ecm-explorer/SystemView</string>
          </field>
          <field name="tabList">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.cms.views.ViewConfig$Tab">
                  <field name="tabName">
                    <string>Info</string>
                  </field>
                  <field name="buttons">
                    <string>viewNodeType; viewPermissions; viewProperties; showJCRStructure</string>
                  </field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>System Template</name>
        <description>Template for display documents in list style</description>
        <object type="org.exoplatform.services.cms.views.TemplateConfig">
          <field name="type">
```

```
            <string>ecmExplorerTemplate</string>
          </field>
          <field name="name">
            <string>SystemView</string>
          </field>
          <field name="warPath">
            <string>/ecm-explorer/SystemView.gtmpl</string>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

2.7. Templates

2.7.1. Application template

A template is an definition of presentation while displaying the saved information

Template service is responsibility for select the right template corresponding to the saved content.

2.7.1.1. Available configuration

2.7.1.1.1. init-params

Configuration name	Data type	Default Value	Description
autoCreateInNewRepository	`Boolean`	true	Allow the application automatically import predefined templated at start up of template service
storedLocation	`String`	war:/conf/dms-extension/dms-artifacts/templates	Location of stored templates
repository	`String`	repository	Location of stored templates
object-param	`Structure`		Configuration for each instance.

2.7.1.1.2. object-param

Configuration name	Data type	Default Value	Description
name	String	template.configuration	The name of this configuration
description	String	configuration for the localtion of templates to inject in jcr	Description of template instance
object-type	Structure		Details configuration for

Configuration name	Data type	Default Value	Description
			each instance type

Object type define all available template file, using the "collection type" configuration

2.7.1.1.3. object

type: name of each object type, it means the type of template, the further configurations for this type are define by the some specify fileds.

Field name	Data type	Description
nodetypeName	`String`	The name of template that is saved as a node in system
documentTemplate	Boolean	Determine if the node type is a document type
label	String	Visual displaying of the title for this node

There are 3 further information related to presentation of each template, that is referencedView: Determine how to display to view. referencedDialog: Determine how to display a diaglog to input information. referencedSkin: Determine the style sheet for displaying.

Each type include some definition as an object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template"with the fields as the properties.

Field name	Data type	Desciption
templateFile	String	The location of the file store for the template's presentation
roles	String	Determine who can access this object (View/Dialog/CSS)

2.7.1.2. Example of configuration for an template

This bellow example is configuration for the nt:file template, any other template will be put in the same level with this template start from the line <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$NodeType"> as the another NodeType.

```
<external-component-plugins>
  <target-component>org.exoplatform.services.cms.templates.TemplateService</target-component>
  <component-plugin>
    <name>addTemplates</name>
    <set-method>addTemplates</set-method>
    <type>org.exoplatform.services.cms.templates.impl.TemplatePlugin</type>
    <init-params>
      <value-param>
        <name>autoCreateInNewRepository</name>
        <value>true</value>
      </value-param>
      <value-param>
```

```
        <name>storedLocation</name>
        <value>war:/conf/dms-extension/dms/artifacts/templates</value>
      </value-param>
      <value-param>
        <name>repository</name>
        <value>repository</value>
      </value-param>
      <object-param>
        <name>template.configuration</name>
        <description>configuration for the localtion of templates to inject in jcr</description>
        <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig">
          <field name="nodeTypes">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$NodeType">
                  <field name="nodetypeName">
                    <string>nt:file</string>
                  </field>
                  <field name="documentTemplate">
                    <boolean>true</boolean>
                  </field>
                  <field name="label">
                    <string>File</string>
                  </field>
                  <field name="referencedView">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template">
                          <field name="templateFile">
                            <string>/file/views/view1.gtmpl</string>
                          </field>
                          <field name="roles">
                            <string>*</string>
                          </field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template">
                          <field name="templateFile">
                            <string>/file/views/admin_view.gtmpl</string>
                          </field>
                          <field name="roles">
                            <string>*:/platform/administrators</string>
                          </field>
                        </object>
                      </value>
                    </collection>
                  </field>
                  <field name="referencedDialog">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template">
                          <field name="templateFile">
                            <string>/file/dialogs/dialog1.gtmpl</string>
                          </field>
                          <field name="roles">
                            <string>*</string>
                          </field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template">
                          <field name="templateFile">
                            <string>/file/dialogs/admin_dialog.gtmpl</string>
                          </field>
                          <field name="roles">
                            <string>*:/platform/administrators</string>
                          </field>
                        </object>
                      </value>
                    </collection>
                  </field>
                  <field name="referencedSkin">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template">
                          <field name="templateFile">
                            <string>/file/skins/Stylesheet-lt.css</string>
                          </field>
```

```
                        <field name="roles">
                          <string>*</string>
                        </field>
                      </object>
                    </value>
                    <value>
                      <object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template">
                        <field name="templateFile">
                          <string>/file/skins/Stylesheet-rt.css</string>
                        </field>
                        <field name="roles">
                          <string>*</string>
                        </field>
                      </object>
                    </value>
                  </collection>
                </field>
              </object>
            </value>
          </collection>
        </field>
      </object>
    </object-param>
  </init-params>
 </component-plugin>
</external-component-plugins>
```

2.7.2. Nodetype template

2.8. Views

A view can include many object parameters. Parameters are used to create default views, templates and actions of Manage view service. View allow administrators to customize many sort of views that can impact to users in exploring workspace. Each object-param have a type that is a class representing all properties of a view.

2.8.1. Fields

org.exoplatform.services.cms.views.ViewConfig

Name	Type	Description
name	`String`	view name, must be unique inside WCM
permissions	`String`	Visibility of the view based on eXo Rights
	example	`*:/platform/administrators`
template	`String`	Specify path to the template location
tabList	`java.util.ArrayList`	include a set of view names.

org.exoplatform.services.cms.views.ViewConfig$Tab

Name	Type	Description
tabName	`String`	tab name that is not uniquely
button	`String`	These have to specify a set of view component names
	example	viewNodeType; viewPermissions; viewProperties; showJCRStructure

org.exoplatform.services.cms.views.TemplateConfig

Name	Type	Description
type	`String`	Specify to a name that is truly a class representing all properties of a view.
name	`String`	These have to specify a set of view component names
warPath	`String`	Specify to a template location for viewing
	example	/ecm-explorer/SystemView.gtmpl

2.8.2. Example of configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.cms.views.ManageViewService</target-component>
  <component-plugin>
    <name>manage.view.plugin</name>
    <set-method>setManageViewPlugin</set-method>
    <type>org.exoplatform.services.cms.views.impl.ManageViewPlugin</type>
    <description>this plugin manage user view</description>
    <init-params>
      <value-param>
        <name>autoCreateInNewRepository</name>
        <value>true</value>
      </value-param>
      <value-param>
        <name>predefinedViewsLocation</name>
        <value>war:/conf/dms-extension/dms/artifacts</value>
      </value-param>
      <value-param>
        <name>repository</name>
        <value>repository</value>
      </value-param>
      <object-param>
        <name>System-View</name>
        <description>View configuration for System workspace</description>
        <object type="org.exoplatform.services.cms.views.ViewConfig">
          <field name="name">
            <string>system-view</string>
          </field>
          <field name="permissions">
            <string>*:/platform/administrators</string>
          </field>
          <field name="template">
            <string>/exo:ecm/views/templates/ecm-explorer/SystemView</string>
          </field>
          <field name="tabList">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.cms.views.ViewConfig$Tab">
```

```
                  <field name="tabName">
                    <string>Info</string>
                  </field>
                  <field name="buttons">
                    <string>viewNodeType; viewPermissions; viewProperties; showJCRStructure</string>
                  </field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>System Template</name>
        <description>Template for display documents in list style</description>
        <object type="org.exoplatform.services.cms.views.TemplateConfig">
          <field name="type">
            <string>ecmExplorerTemplate</string>
          </field>
          <field name="name">
            <string>SystemView</string>
          </field>
          <field name="warPath">
            <string>/ecm-explorer/SystemView.gtmpl</string>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

2.9. Extensions

Chapter 3. Inside WCM Templates

3.1. Content types

Overview

The templates are applied to a NodeType or a metadata MixinType. Two kinds of templates exist :

- **dialogs** : are html forms that allow to create node instances
- **views** : are html fragments used to display nodes

From the ECM admin portlet, *Manage Template* lists existing NodeTypes that have been associated to Dialog and/or View templates. These templates can be attached to permissions (in the usual *membership:group_ form), so that a specific one is displayed according to the rights of the user (very useful in a content validation workflow activity).*

DocumentType

1. checkbox allows to say if the nodetype should be considered as a **DocumentType**. File Explorer considers such nodes as user content and applies the following behavior :

- View template will be used to display the DocumentType nodes
- DocumentTypes nodes can created by the '*Add Document*' action
- non DocumentType are hidden (unless *'Show non document types'* option is checked)

Templates are written using Groovy Templates and will require some experience with JCR API and HTML notions.

3.1.1. Dialog

Dialogs are groovy templates that generate forms by mixing static HTML fragments and groovy calls to the components responsible for building the UI at runtime. The result is a simple but powerfull syntax.

3.1.1.1. Text Field

3.1.1.1.1. Parameters

		Text Field
jcrPath	`Path`	relative path inside the current node
	example	`jcrPath=/node/exo:title`
mixintype	`List of String`	List of the mixin types we want to initialize when creating the content

		Text Field
	example	`mixintype=mix:votable,mix:commentable,mi`
		`validate=empty,name`
validate	`List of String`	use some validators. Possible values are : empty, name, datetime.
	example	`validate=empty`
		`validate=empty,name`
editable	`String`	Text input will be editable if value is *if-null*
	example	`editable=if-null`
multiValues	`Boolean`	Show a multi values component if true. Must be used only with corresponding multi-values properties
	example	`multiValues=true`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`

Warning

Note that *mixintype* can be used only in the root node field (commonly known as the name field)

3.1.1.1.2. Example

```
<%
  String[] fieldTitle = ["jcrPath=/node/exo:title", "validate=empty"] ;
  uicomponent.addTextField("title", fieldTitle) ;
%>
```

3.1.1.2. Hidden Field

`TODO : DOC HERE`

3.1.1.3. Text Area Field

3.1.1.3.1. Parameters

		Text Field
jcrPath	`Path`	relative path inside the current node
	example	`jcrPath=/node/exo:description`
mixintype	`List of String`	List of the mixin types we want to

		Text Field
		initialize when creating the content
	example	`mixintype=mix:votable,mix:commentable,mi`
		`validate=empty,name`
validate	`List of String`	use some validators. Possible values are : empty, name, datetime.
	example	`validate=empty`
		`validate=empty,name`
editable	`String`	Text input will be editable if value is *if-null*
	example	`editable=if-null`
multiValues	`Boolean`	Show a multi values component if true. Must be used only with corresponding multi-values properties
	example	`multiValues=true`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`
rows	Number	The initial textarea's number of rows
cols	Number	The initial textarea's number of cols

3.1.1.3.2. Example

```
<%
String[] fieldDescription = ["jcrPath=/node/exo:description", "validate=empty"] ;
uicomponent.addTextAreaField("description", fieldDescription)
%>
```

3.1.1.4. Rich Text Field

3.1.1.4.1. Parameters

jcrPath	`Path`	relative path inside the current node	
	example	`jcrPath=/node/exo:title`	
validate	`List of String`	use some validators. Possible values are : empty, name, datetime.	

		Text Field	
	example	`validate=empty`	
multiValues	`Boolean`	Show a multi values component if true. Must be used only with corresponding multi-values properties	
	example	`multiValues=true`	
options	`String`	input options of rich text field	
	example	`options=basic;width:'100%';height:200px;`	

3.1.1.4.2. Example

```
<%
String[] fieldSummary = ["jcrPath=/node/exo:summary", "options=toolbar:BasicWCM,width:'100%',height:200px", "val
uicomponent.addRichtextField("summary", fieldSummary) ;
%>
```

3.1.1.5. Calendar Field

3.1.1.5.1. Parameters

		Calendar Field
jcrPath	`Path`	relative path inside the current node
	example	`jcrPath=/node/exo:publishedDate`
options	`List of String`	use for format data. Possible values are : displaytime, displaydatetime.
	example	`options=displaytime`
		`options=displaydatetime`
validate	`List of String`	use some validators. Possible values are : empty, datetime.
	example	`validate=empty`
		`validate=datetime`
editable	`String`	Text input will be editable if value is *if-null*
	example	`editable=if-null`
multiValues	`Boolean`	Show a multi values component if true. Must be used only with

		Calendar Field
		corresponding multi-values properties
	example	`multiValues=true`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`

3.1.1.5.2. Example

```
<%
  String[] fieldPublishedDate = ["jcrPath=/node/exo:publishedDate", "options=displaytime", "validate=datetime",
  uicomponent.addCalendarField("publishedDate", fieldPublishedDate) ;
%>
```

3.1.1.6. Upload Field

3.1.1.6.1. Parameters

		Upload Field
jcrPath	`Path`	relative path inside the current node
		`validate=empty,name`
validate	`List of String`	use some validators. Allowed values are : empty, name, datetime.
	example	`validate=empty`
editable	`String`	Text input will be editable if value is *if-null*
	example	`editable=if-null`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`

3.1.1.6.2. Example

```
<%
  String[] fieldMedia = ["jcrPath=/node/jcr:content/jcr:data"] ;
  uicomponent.addUploadField("media", fieldMedia) ;
%>
```

3.1.1.7. Radio Field

3.1.1.7.1. Parameters

		Radio Field
jcrPath	`Path`	relative path inside the current node
		`validate=empty,name`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`

3.1.1.7.2. Example

```
<%
  String[] fieldDeep = ["jcrPath=/node/exo:isDeep", "defaultValues=true"];
  uicomponent.addRadioBoxField("isDeep", fieldDeep);
%>
```

3.1.1.8. SelectBox Field

`TODO : DOC HERE`

3.1.1.9. CheckBox Field

3.1.1.9.1. Parameters

		CheckBox Field
jcrPath	`Path`	relative path inside the current node
	example	`jcrPath=/node/exo:isDeep`
validate	`List of String`	use some validators. Possible values are : empty
	example	`validate=empty`
		`validate=empty`
editable	`String`	Text input will be editable if value is *if-null*
	example	`editable=if-null`
multiValues	`Boolean`	Show a multi values component if true. Must be used only with corresponding multi-values properties
	example	`multiValues=true`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`

3.1.1.9.2. Example

```
<%
  String[] fieldDeep = ["jcrPath=/node/exo:isDeep", "defaultValues=true"];
  uicomponent.addCheckBoxField("isDeep", fieldDeep);
%>
```

3.1.1.10. Interceptors

TO add an intercepter into a dialog, we can use this method **`uicomponent.addInterceptor(String scriptPath, String type)`**

		Parameters
scriptPath	`String`	The relative path to the script file
type	`String`	The type of intercepter: `prev` or `post`

3.1.1.10.1. Example

```
<%
  uicomponent.addInterceptor("ecm-explorer/interceptor/PreNodeSaveInterceptor.groovy", "prev");
%>
```

3.1.1.11. Mixin Field

3.1.1.11.1. Parameters

		add Mixin Field
jcrPath	`Path`	relative path inside the current node
	example	`jcrPath=/node/exo:name`
editable	`String`	Text input will be editable if value is *if-null*
	example	`editable=if-null`
visible	`Boolean`	Input will be visible if true
	example	`visible=true`

3.1.1.11.2. Example

```
<%
        String[] fieldId = ["jcrPath=/node", "editable=false", "visible=if-not-null"] ;
  uicomponent.addMixinField("id", fieldId) ;
%>
```

3.1.2. View

```
<%
        def node = uicomponent.getNode() ;
        def originalNode = uicomponent.getOriginalNode()
%>
```

```
<%=node.getName()%>
```

```
<%if(node.hasProperty("exo:title")) {%>
  <%=node.getProperty("exo:title").getString()%>
<%}%>
```

```
<%
        import java.text.SimpleDateFormat ;
        SimpleDateFormat dateFormat = new SimpleDateFormat() ;
%>
...

<%if(node.hasProperty("exo:date")) {
  dateFormat.applyPattern("MMMMM dd yyyy") ;
%>
  <%=dateFormat.format(node.getProperty("exo:date").getDate().getTime())%>
<%}%>
```

```
<%=_ctx.appRes("Sample.view.label.node-name")%>
```

3.2. List of Contents

3.2.1. Folder based Template

3.2.2. Category tree Template

Chapter 4. Inside WCM Explorer

4.1. CSS

4.2. Javascript

4.3. CKEditor

Chapter 5. Extensions

5.1. REST Services

5.2. UI Extensions

Chapter 6. JMX Events

Chapter 7. Java Services

7.1. TaxonomyService

7.2. LinkManager

7.3. PublicationManager

7.4. WCMComposer

Chapter 8. FAQ

8.1.1. How do I create a new Site ?

8.1.2. How do I create a new Content Type ?

8.1.3. How do I create a new Site ?