eXo Platform 3.0 - Content Functions

1. Preface 1
1.1. Get Started with eXo Content 1
1.2. Package 1
2. Applications 3
2.1. Portlets 3
2.1.1. Content Detail 3
2.1.2. Content List 4
2.1.3. Search 8
2.1.4. Content Explorer 10
2.1.5. Administration 13
2.1.6. Fast Content Creator 13
2.1.7. Form Builder 15
2.1.8. Authoring 15
2.1.9. Newsletter 16
3. Configuration 17
3.1. Components 17
3.1.1. ActionServiceContainer 17
3.1.2. ApplicationTemplateManagerService 17
3.1.3. FragmentCacheService 18
3.1.4. JodConverterService 18
3.1.5. LiveLinkManagerService 19
3.1.6. LockService 19
3.1.7. NewsletterInitializationService 19
3.1.8. NewsletterManagerService 22
3.1.9. QueryService 22
3.1.10. TaxonomyService 23
3.1.11. ThumbnailService 24
3.1.12. TimelineService 25
3.1.13. WatchDocumentService 25
3.1.14. WCMService 26
3.2. External Component Plugins 27
3.2.1. AuthoringPublicationPlugin 27
3.2.2. AddNameSpacesPlugin 29
3.2.3. BaseActionPlugin 30
3.2.4. ContentTypeFilterPlugin 30
3.2.5. ContextPlugin 31
3.2.6. CreatePortalPlugin 32
3.2.7. FriendlyPlugin 32
3.2.8. ImageThumbnailPlugin 33
3.2.9. InitialWebcontentPlugin 34
3.2.10. LinkDeploymentPlugin 37
3.2.11. LockGroupsOrUsersPlugin 38
3.2.12. ManageDrivePlugin 39
3.2.13. ManageViewPlugin 41
3.2.14. PDFThumbnailPlugin 43
3.2.15. PorletTemplatePlugin 44
3.2.16. PredefinedProcessesPlugin 47
3.2.17. PublicationPlugin 48
3.2.18. QueryPlugin 48
3.2.19. RemovePortalPlugin 50
3.2.20. ScriptActionPlugin 50
3.2.21. ScriptPlugin 50

3.2.22. StageAndVersionPublicationPlugin 52
3.2.23. TagPermissionPlugin 55
3.2.24. TagStylePlugin 55
3.2.25. TaxonomyPlugin 57
3.2.26. TemplatePlugin 61
3.2.27. WCMPublicationPlugin 64
3.2.28. XMLdeploymentPlugin 67
4. Developer references 70
4.1. WCM Templates 70
4.1.1. Content types 70
4.1.2. List of Contents 79
4.2. WCM Explorer 80
4.2.1. CSS 80
4.2.2. CKEditor 81
4.3. Extensions 81
4.3.1. REST Services 81
4.3.2. UI Extensions 83
4.3.3. Authoring Extension 88
4.4. Public REST APIs 96
4.4.1. ThumbnailRESTService 97
4.4.2. RssConnector 98
4.4.3. FCKCoreRESTConnector 99
4.4.4. ResourceBundleConnector 100
4.4.5. VoteConnector 101
4.4.6. DriverConnector 102
4.4.7. GadgetConnector 103
4.4.8. PortalLinkConnector 103
4.4.9. GetEditedDocumentRESTService 104
4.4.10. PublicationGetDocumentRESTService 104
4.4.11. FavoriteRESTService 105
4.4.12. RESTImagesRendererService 106
4.4.13. LifecycleConnector 106
4.4.14. CopyContentFile 107
4.4.15. PDFViewerRESTService 108
4.5. Public Java APIs 108
4.5.1. TaxonomyService 108
4.5.2. LinkManager 119
4.5.3. PublicationManager 124
4.5.4. WCMComposer 125
4.5.5. NewFolksonomy 127
4.5.6. ApplicationTemplateManager 137
4.5.7. NodeFinder 140
4.5.8. JodConverter 146
4.6. FAQ 147
4.6.1. How to deploy a workflow? 147

Chapter 1. Preface

1.1. Get Started with eXo Content

eXo Content is a fully integrated content management solution inside eXo Platform. Basically, in eXo Platform, the extension mechanism is used to add content features. If you want more information about the extension mechanism, you can read the GateIn documentation:

http://docs.jboss.com/gatein/portal/3.1.0-FINAL/reference-guide/en-US/html/index.html

This integrated solution provides users with a comprehensive platform for integrating applications, managing and publishing content - all from a single, familiar console.

When starting a new project, you can see eXo Content as a eXo Content Java developer Platform. In this reference guide, we will give you guidelines related to building stable applications using eXo Content from the inside.

1.2. Package

All package actions in eXo Content are started from the `delivery` folder, so you should go to this folder first.

```
$ cd ecms/delivery
```

- Package WCM standalone version - Tomcat bundle

```
$ cd wcm/assembly
$ mvn clean install
```

- Package WCM with workflow enabled - Tomcat bundle

```
$ cd wkf-wcm/assembly
$ mvn clean install
```

- Make WCM EAR packages to run with jBoss

1. Make WCM extension EAR

```
$ cd packaging/wcm/ear
$ mvn clean install
```

2. Make WCM demo sites EAR

```
$ cd packaging/ecmdemo/ear
$ mvn clean install
```

```
```

3. Make workflow EAR

```
$ cd packaging/workflow/ear
$ mvn clean install
```

Chapter 2. Applications

2.1. Portlets

2.1.1. Content Detail

The Content Detail portlet is packaged in the *presentation.war* file. The portlet allows users to view the detail of a specific content.

2.1.1.1. Available preferences

Preference	Type	Default Value	Description
repository	`String`	`repository`	The repository where data are stored and maintained.
workspace	`String`	`collaboration`	The workspace where 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 the top of the portlet.
ShowDate	`Boolean`	`false`	Show the content date on the top of the portlet.
ShowOptionBar	`Boolean`	`false`	Show a bar with some actions (Print, Back).
ContextEnable	`Boolean`	`false`	Define if the portlet will use the parameter on URL as the path to content to display or not.
ParameterName	`String`	`content-id`	Define which 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 (if any).
sharedCache	`Boolean`	`true`	Define if the portlet will cache the displayed contents.

2.1.1.2. 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>
  <preference>
    <name>sharedCache</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

Note

In WCM 2.2.0, some preferences are no longer used, for example, the ShowPrintAction preference.

2.1.2. Content List

The Content List portlet is packaged in the *presentation.war* file. It shows a list of contents.

2.1.2.1. Available preferences

Preference	Type	Default Value	Description
mode	`String`	`AutoViewerMode`	The mode that the portlet uses to display content: all contents in a specific folder or all specific contents 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 the portlet are sorted.
orderType	`String`	`DESC`	The type of the content sort method: ascending or descending
header	`String`		The 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/content-list-template/UICont`	The path to the template used to display the contents in this portlet.
paginatorTemplatePath	`String`	`/exo:ecm/views/templates/content-list-paginators/`	The path to the paginator used to display the contents in this portlet.
itemsPerPage	`Integer`	10	The 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.

Preference	Type	Default Value	Description
showHeader	`Boolean`	`true`	This value indicates whether the content 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 or not.(After clicking this button, the 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 the 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 parameter name

Preference	Type	Default Value	Description
			which shows the folder path in URL when the "Read more" button is clicked.
showClvBy	`String`	`folder-id`	The parameter name which shows the folder path in URL.
sharedCache	`Boolean`	`true`	Define if the portlet will cache the displayed contents.

2.1.2.2. 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/UIContentListPresentationDefault.gtmpl</value>
    <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>
  <preference>
    <name>sharedCache</name>
    <value>true</value>
    <read-only>false</read-only>
  </preference>
</portlet-preferences>
```

2.1.3. Search

The Search portlet is packaged in the *searches.war* file. It allows users to do a search with any string. In WCM 2.2.0, there are three types of search: quick search, advanced search and search with saved queries.

2.1.3.1. Available preferences

Preference	Type	Default value	Description
repository	`String`	`repository`	The repository is a place where data are stored and maintained.
workspace	`String`	`collaboration`	The workspace where the content is stored.
searchFormTemplatePath	`String`	`/exo:ecm/views/templates/WCM Advance Search/search-form/UIDefaultSearchForm.gtmpl`	The path to the search form template.
searchResultTemplatePath	`String`	`/exo:ecm/views/templates/WCM Advance Search/search-result/UIDefaultSearchResult.gtmpl`	The path to the search result template.
searchPaginatorTemplatePath	`String`	`/exo:ecm/views/templates/WCM Advance Search/search-paginator/UIDefaultSearchPaginator.gtmpl`	The path to the search paginator template.
searchPageLayoutTemplatePath	`String`	`/exo:ecm/views/templates/WCM Advance Search/search-page-layout/UISearchPageLayoutDefault.gtmp`	The 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 is used to display the search result.

2.1.3.2. 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>
```

2.1.4. Content Explorer

The Content Explorer portlet is packaged in the *ecmexplorer.war* file. It is used to manage all documents in different drives. With this portlet, users can do many different actions depending on their roles, such as adding/deleting a category and a document, showing/hiding a node, managing publication and more.

2.1.4.1. Available preferences

Preference	Type	Default value	Description
repository	`String`	`repository`	The repository name which is used in an instance of Content Explorer.
workspace	`String`	N/A	Not in use. The workspace name was included in the Drive.
path	`String`	N/A	The path of the node. This preference will be used when the usecase chosen is Parameterize.
drive	`String`	N/A	Not in use. Replaced by driveName preference.
views	`String`	N/A	Not in use. The views will be displayed basing on the Drive which a user has the access permission.
allowCreateFolders	`String`	N/A	Allow to create a folder by type. When you do not specify the value, then the default value will be nt:unstructured, nt:folder.

Preference	Type	Default value	Description
categoryMandatoryWhenFileUpload	Boolean	false	Force a user to add a category when uploading or creating a document.
uploadFileSizeLimitMB	`Float`	150	The maximum size of a file that is uploaded to the system (MB).
usecase	`String`	`selection`	The behavior to access Content Explorer. By default, the "selection" option is configured. Besides "selection", there are four other ways to configure the Content Explorer: **Jailed**,**Personal**,**Social**,**Parameterize**.
driveName	`String`	`Private`	The name of a drive that a user wants to access.
trashHomeNodePath	`String`	`/Trash`	The location to store the deleted nodes.
trashRepository	`String`	`repository`	The name of the repository where stores the deleted nodes.
trashWorkspace	`String`	`collaboration`	The name of the workspace where stores the deleted nodes.
editInNewWindow	`Boolean`	`false`	Allow editing documents with or without a window popup.
showTopBar	`Boolean`	`true`	Allow showing the Top bar or not.
showActionBar	`Boolean`	`true`	Allow showing the Action bar or not.
showSideBar	`Boolean`	`true`	Allow showing the Side bar or not.
showFilterBar	`Boolean`	`true`	Allow showing the Filter bar or not.

2.1.4.2. Sample 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>
```

2.1.5. Administration

The Administration portlet is packaged in the *ecmadmin.war* file. The portlet manages the main ECM functions, including categories and tags, content presentation, types of content and advanced configuration.

2.1.5.1. Available preferences

Preference	Type	Default	Description
repository	`String`	`Repository`	The name of the current repository.

2.1.5.2. Sample Configuration

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

2.1.6. Fast Content Creator

The Fast Content Creator portlet is packaged in the *formgenerator.war* file. The portlet consists of two modes: Standard Content Creator and Basic Content Creator which allows users to quickly create a content without accessing the Content Explorer portlet.

2.1.6.1. Available preferences

Preference	Type	Default Value	Description
mode	`String`	`basic`	The default mode for **Fast Content Creator** portlet.
repository	`String`	`repository`	The name of the current repository.
workspace	`String`	`collaboration`	The workspace where the content is stored.
path	`String`	`/Groups/platform/users/Documents`	The destination path where the content is

Preference	Type	Default Value	Description
			stored.
type	`String`	`exo:article`	The node type of document which is shown on the dialog form.
saveButton	`String`	`Save`	The custom button: **Save**.
saveMessage	`String`	`This node has been saved successfully`	The custom message when the user clicks the **Save** button.
isRedirect	`Boolean`	`false`	Specify whether redirecting to another page or not.
redirectPath	`String`	`http://www.google.com.vn/#`	The path to which the page will redirect.
isActionNeeded	`Boolean`	`true`	Specify whether an action is needed to save to the configuration or not.

2.1.6.2. Sample 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>
```

2.1.7. Form Builder

The Form Builder portlet is packaged in the *formgenerator.war* file. It allows users to create node types and document templates for node types.

2.1.7.1. Available preferences

Preference	Type	Default Value	Description
repository	`String`	`repository`	The current repository name which is always "repository".

2.1.7.2. Sample Configuration

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

2.1.8. Authoring

The Authoring portlet is packaged in the *authoring-apps.war* file. The portlet allows users to manage contents in draft and contents needed to be approved or published.

2.1.8.1. Available preferences

Preference	Type	Default	Description
repository	`String`	`Repository`	The name of the repository.
workspace	`String`	`Collaboration`	The name of the workspace.
drive	`String`	`Collaboration`	The name of the drive.

2.1.8.2. Sample Configuration

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

2.1.9. Newsletter

The Newsletter portlet is packaged in the *newsletter.war* file. The portlet is used to help users quickly get the updated newsletter from a site.

Chapter 3. Configuration

3.1. Components

3.1.1. ActionServiceContainer

The *ActionServiceContainer* component is used to manage actions (add actions, remove actions, execute actions and more) in the system. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
          <key>org.exoplatform.services.cms.actions.ActionServiceContainer</key>
          <type>org.exoplatform.services.cms.actions.impl.ActionServiceContainerImpl</type>
          <init-params>
      <value-param>
        <name>workspace</name>
        <value>system</value>
      </value-param>
      <value-param>
        <name>repository</name>
        <value>repository</value>
      </value-param>
      </init-params>
</component>
```

Details:

value-param	Possible value	Default value	Description
workspace	String	system	The workspace name.
repository	String	repository	The repository name.

3.1.2. ApplicationTemplateManagerService

The ApplicationTemplateManagerService component is used to manage dynamic Groovy templates for WCM-based products. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
    <key>org.exoplatform.services.cms.views.ApplicationTemplateManagerService</key>
    <type>org.exoplatform.services.cms.views.impl.ApplicationTemplateManagerServiceImpl</type>
    <init-params>
        <properties-param>
        <name>storedLocations</name>
                <property name="repository" value="system"/>
        </properties-param>
      </init-params>
  </component>
```

Details:

Properties-param	Property name	Possible value	Default value	Description
storedLocations	repository	string	system	The repository

Properties-param	Property name	Possible value	Default value	Description
				name.

3.1.3. FragmentCacheService

```
<component>
    <key>org.exoplatform.services.portletcache.FragmentCacheService</key>
    <type>org.exoplatform.services.portletcache.FragmentCacheService</type>
    <init-params>
      <value-param>
        <name>cleanup-cache</name>
        <description>The cleanup cache period in seconds</description>
        <value>300</value>
      </value-param>
    </init-params>
</component>
```

Details:

Value-param	Possible value	Default value	Description
cleanup-cache	integer	300	

3.1.4. JodConverterService

The JodConverterServices component is used to convert documents into different office formats. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
          <key>org.exoplatform.services.cms.jodconverter.JodConverterService</key>
          <type>org.exoplatform.services.cms.jodconverter.impl.JodConverterServiceImpl</type>
          <init-params>
      <value-param>
        <name>host</name>
        <value>127.0.0.1</value>
      </value-param>
      <value-param>
        <name>port</name>
        <value>8100</value>
      </value-param>
    </init-params>
</component>
```

Details:

Value-param	Possible value	Default value	Description
host	The host IP	127.0.0.1	The host from which a document is converted into different office formats.
port	The port number	8100	The port number is open to accept converting a

Value-param	Possible value	Default value	Description
			document into different office formats.

3.1.5. LiveLinkManagerService

The LiveLinkManagerService component is used to check broken links, update links when the links are edited and extract links to return a list of all links. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/wcm-configuration.xml*.

```
<component>
    <key>org.exoplatform.services.wcm.link.LiveLinkManagerService</key>
    <type>org.exoplatform.services.wcm.link.LiveLinkManagerServiceImpl</type>
    <init-params>
      <properties-param>
        <name>server.config</name>
        <description>server.address</description>
        <property name="scheme" value="http" />
        <property name="hostName" value="localhost" />
        <property name="port" value="8080" />
      </properties-param>
    </init-params>
 </component>
```

Details:

Properties-param	Property name	Possible value	Default value	Description
server.config	scheme hostName port	http/https string the port number	http localhost 8080	All the property names are used together to configure the server. Here is an example about the server configuration: http//:localhost:8080.

3.1.6. LockService

The LockService component is used to manage all locked nodes and allows unlocking the locked nodes in the system. It is also used to assign the Lock right to a user or a user group or a membership. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
    <key>org.exoplatform.services.cms.lock.LockService</key>
    <type>org.exoplatform.services.cms.lock.impl.LockServiceImpl</type>
 </component>
```

3.1.7. NewsletterInitializationService

The NewsletterInitializationService component is used to initiate some data for the newsletter portlet. The configuration of this component is found in *packaging/ecmdemo/webapp/src/main/webapp/WEB-INF/conf/sample-portal/wcm/newsletter-configuration.xml*.

```
<component>
  <type>org.exoplatform.services.wcm.newsletter.NewsletterInitializationService</type>
  <init-params>
    <values-param>
      <name>portalNames</name>
      <value>classic</value>
      <value>acme</value>
    </values-param>
    <values-param>
      <name>administrators</name>
      <value>root</value>
      <value>john</value>
    </values-param>
    <object-param>
      <name>marketing</name>
      <description>marketing</description>
      <object type="org.exoplatform.services.wcm.newsletter.NewsletterCategoryConfig">
        <field name="name">
          <string>marketing</string>
        </field>
        <field name="title">
          <string>Marketing</string>
        </field>
        <field name="description">
          <string>You want to know where we are, where we go ?</string>
        </field>
        <field name="moderator">
          <string>*:/platform/web-contributors</string>
        </field>
      </object>
    </object-param>
    <object-param>
      <name>general</name>
      <description>general</description>
      <object type="org.exoplatform.services.wcm.newsletter.NewsletterCategoryConfig">
        <field name="name">
          <string>general</string>
        </field>
        <field name="title">
          <string>General</string>
        </field>
        <field name="description">
          <string>General information about us</string>
        </field>
        <field name="moderator">
          <string>*:/platform/web-contributors</string>
        </field>
      </object>
    </object-param>
    <object-param>
      <name>subscription2</name>
      <description>subscription2</description>
      <object type="org.exoplatform.services.wcm.newsletter.NewsletterSubscriptionConfig">
        <field name="name">
          <string>results</string>
        </field>
        <field name="title">
          <string>Results</string>
        </field>
        <field name="description">
          <string>Monthly newsletter about our results and forecasts</string>
        </field>
        <field name="categoryName">
          <string>general</string>
        </field>
        <field name="redactor">
          <string>*:/platform/web-contributors</string>
        </field>
      </object>
    </object-param>
    <object-param>
      <name>subscription1</name>
```

```
        <description>subscription1</description>
        <object type="org.exoplatform.services.wcm.newsletter.NewsletterSubscriptionConfig">
          <field name="name">
            <string>checklist</string>
          </field>
          <field name="title">
            <string>Check-List</string>
          </field>
          <field name="description">
            <string>Weekly newsletter with general topics</string>
          </field>
          <field name="categoryName">
            <string>general</string>
          </field>
          <field name="redactor">
            <string>*:/platform/web-contributors</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>subscription3</name>
        <description>subscription3</description>
        <object type="org.exoplatform.services.wcm.newsletter.NewsletterSubscriptionConfig">
          <field name="name">
            <string>market</string>
          </field>
          <field name="title">
            <string>The market</string>
          </field>
          <field name="description">
            <string>What's on the market today ?</string>
          </field>
          <field name="categoryName">
            <string>marketing</string>
          </field>
          <field name="redactor">
            <string>*:/platform/web-contributors</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>user1@gmail.com</name>
        <description>user1@gmail.com</description>
        <object type="org.exoplatform.services.wcm.newsletter.config.NewsletterUserConfig">
          <field name="mail">
            <string>user1@gmail.com</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>user2@gmail.com</name>
        <description>user2@gmail.com</description>
        <object type="org.exoplatform.services.wcm.newsletter.config.NewsletterUserConfig">
          <field name="mail">
            <string>user2@gmail.com</string>
          </field>
        </object>
      </object-param>
    </init-params>
  </component>
```

Details:

Value-param	Possible value	Default value	Description
portalNames	string	classic, acme	The portal names.
administrators	string	root	The administrator who manages the newsletter portlet.
name	string		The name of categories or subscriptions.

Value-param	Possible value	Default value	Description
title	string		The title of categories or subscriptions.
moderator	string	*:/platform/web-contributors	The users or groups that can moderate the newsletter portlet.
categoryName	string	general, marketing	Thu categories name.
redactor	string	*:/platform/web-contributors	The users or groups that are newsletter redactor.
mail	string		The email that user uses to subscribe.

3.1.8. NewsletterManagerService

The NewsletterManagerService component is used to send newsletters to subscribers. The configuration of this component is found in *core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/ext-newsletter-configuration.xml*.

```
<component>
    <type>org.exoplatform.services.wcm.newsletter.NewsletterManagerService</type>
    <init-params>
      <value-param>
        <name>repository</name>
        <value>repository</value>
      </value-param>
      <value-param>
        <name>workspace</name>
        <value>collaboration</value>
      </value-param>
    </init-params>
</component>
```

Details:

Value-param	Possible value	Default value	Description
repository	string	repository	The repository name.
workspace	string	collaboration	The workspace name.

3.1.9. QueryService

The QueryService component is used to manage many queries, including adding a query, removing a query and executing a query. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
          <key>org.exoplatform.services.cms.queries.QueryService</key>
          <type>org.exoplatform.services.cms.queries.impl.QueryServiceImpl</type>
          <init-params>
            <value-param>
```

```
              <name>workspace</name>
              <value>system</value>
            </value-param>
            <value-param>
              <name>relativePath</name>
              <value>Private/Queries</value>
            </value-param>
            <value-param>
              <name>group</name>
              <value>*:/admin</value>
            </value-param>
          </init-params>
</component>
```

Details:

Value-param	Possible value	Default value	Description
workspace	string	system	The workspace name.
relativePath	Private/Queries	Private/Queries	The path to the query location.
group	string	*:/admin	The group is allowed to access the query folder.

3.1.10. TaxonomyService

The TaxonomyService is used to sort documents to ease searches when browsing documents online. It provides a multi-dimensional set of paths to find a document. 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. Taxonomies are stored in the JCR itself and the JCR Reference functionality is used to provide that advanced WCM feature. The tree of taxonomies can be managed simply, such as copying/cutting/pasting nodes, or adding and removing 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 to the created documents.

```
<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.TaxonomyTreeDefaultUserPermissi
                          <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>
                <value>
                  <object type="org.exoplatform.services.cms.taxonomy.impl.TaxonomyTreeDefaultUserPermission$Per
                    <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>
      </object-param>
          </init-params>

</component>
```

Details: **Object type**: *org.exoplatform.services.cms.taxonomy.impl.TaxonomyTreeDefaultUserPermission*

Field name	Collection type	Description
permissions	`java.util.ArrayList<TaxonomyTreeDefaultUserPermission$Permission>`	The list of the default permissions to access the taxonomy tree.

Object type: *org.exoplatform.services.cms.taxonomy.impl.TaxonomyTreeDefaultUserPermission$Permission*

Field name	Type	Default value	Description
identity	String	*:/platform/administrators	The name of a user or a group.
read	Boolean	true	The permission to read the taxonomy tree.
addNode	Boolean	true	The permission to add a node to the taxonomy tree.
setProperty	Boolean	true	The permission to set properties for a node in the taxonomy tree.
remove	Boolean	true	The permission to remove a node from the taxonomy tree.

3.1.11. ThumbnailService

The ThumbnailService component is used to resize all the images into different sizes. Besides the default sizes, it also allows users to customize the images into the desired sizes. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
          <key>org.exoplatform.services.cms.thumbnail.ThumbnailService</key>
          <type>org.exoplatform.services.cms.thumbnail.impl.ThumbnailServiceImpl</type>
          <init-params>
      <value-param>
              <name>smallSize</name>
              <value>32x32</value>
      </value-param>
      <value-param>
              <name>mediumSize</name>
              <value>64x64</value>
```

```
      </value-param>
      <value-param>
              <name>largeSize</name>
              <value>300x300</value>
      </value-param>
      <value-param>
              <name>enable</name>
              <value>false</value>
      </value-param>
      <value-param>
        <name>mimetypes</name>
        <value>image/jpeg;image/png;image/gif;image/bmp</value>
      </value-param>
    </init-params>
</component>
```

Details:

Value-param	Possible value	Default value	Description
smallSize	32x32	32x32	The small thumbnail size.
mediumSiz	64x64	64x64	The medium thumbnail size.
largeSize	300x300	300x300	The large thumbnail size.
enable	boolean	false	Specify if the thumbnail is displayed or not.
mimetypes	images formats	image/jpeg;image/png;image/gif;image/bmp	The image formats are supported.

3.1.12. TimelineService

The TimelineService component allows documents to display by days, months and years. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
    <key>org.exoplatform.services.cms.timeline.TimelineService</key>
    <type>org.exoplatform.services.cms.timeline.impl.TimelineServiceImpl</type>
    <init-params>
      <value-param>
        <name>itemPerTimeline</name>
        <value>5</value>
      </value-param>
    </init-params>
</component>
```

Details:

Value-param	Possible value	Default value	Description
itemPerTimeline	integer	5	The number of documents are displayed.

3.1.13. WatchDocumentService

WatchDocumentService allows users to watch/unwatch a document. If they are watching the document, they will receive a notification mail when there are any changes on the document. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
          <key>org.exoplatform.services.cms.watch.WatchDocumentService</key>
          <type>org.exoplatform.services.cms.watch.impl.WatchDocumentServiceImpl</type>
          <init-params>
      <object-param>
        <name>messageConfig</name>
        <description>description</description>
        <object type="org.exoplatform.services.cms.watch.impl.MessageConfig">
          <field  name="sender"><string>support@exoplatform.com</string></field>
          <field  name="subject"><string>Your watching document is changed</string></field>
          <field  name="content">
                  <string>The document that you are watching is changed.
                    Please go to ecm to see this change
            </string>
          </field>
        </object>
      </object-param>
    </init-params>
</component>
```

Details:

- Object type: *org.exoplatform.services.cms.watch.impl.MessageConfig*

Field name	Type	Default value	Description
sender	string	support@exoplatform.com	The sender who sends the notification mail.
subject	string	Your watching document is changed.	The subject of the notification mail.
content	string	The document that you are watching is changed. Please go to ecm to see this change.	The content of the notification mail.

3.1.14. WCMService

The WCMService allows setting expiration cache of portlets and checking given portals if they are shared portals or not. It also gets reference contents basing on item identifiers. The configuration of this component is found in */core/core-configuration/src/main/webapp/WEB-INF/conf/wcm-core/core-services-configuration.xml*.

```
<component>
    <key>org.exoplatform.services.wcm.core.WCMService</key>
    <type>org.exoplatform.services.wcm.core.impl.WCMServiceImpl</type>
    <init-params>
      <properties-param>
        <name>server.config</name>
        <description>server.config</description>
        <property name="expirationCache" value="30" />
      </properties-param>
    </init-params>
</component>
```

Details:

Properties-param	Property name	Possible value	Default value	Description
server.config	expirationCache	integer	30	The period in which the cache is clear in second. By default, the cache is cleared every 30 seconds.

3.2. External Component Plugins

3.2.1. AuthoringPublicationPlugin

3.2.1.1. Fields

Object type: *org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.LifecyclesConfig*

Name	Type	Default Value	Description
lifecycles	`array list`	`java.util.ArrayList`	The list of lifecycles.

Object type: *org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.LifecyclesConfig$Lifecycle*

Name	Type	Default Value	Description
name	`String`	`N/A`	The name of the life cycle.
publicationPlugin	`string`	`Authoring publication`	The pubkication lugin name.
states	`array list`	`java.util.ArrayList`	The list of the publication states.

Object type: *org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.LifecyclesConfig$State*

Name	Type	Default Value	Description
state	`string`	`N/A`	The publication states: draft, pending, staged, approved or published.
membership	`string`	`N/A`	The user or group.

Object type: *org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig*

Name	Type	Default Value	Description
contexts	`array list`	`java.util.ArrayList`	The list of contexts.

Object type: *org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig$Context*

Name	Type	Default Value	Description
name	`string`	`contextdefault`	The context name.
priority	`string`	`200`	The context priority, the higher number indicates higher priority. Because a site may have several life cycles so the life cycle with higher priority will be executed.
lifecycle	`string`	`lifecycle1`	The name of lifecycle.

3.2.1.2. Sample configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.wcm.extensions.publication.PublicationManager</target-component>
  <component-plugin>
    <name>AddLifecycle</name>
    <set-method>addLifecycle</set-method>
    <type>org.exoplatform.services.wcm.extensions.publication.lifecycle.StatesLifecyclePlugin</type>
    <description>Configures</description>
    <priority>1</priority>
     <init-params>
      <object-param>
        <name>lifecycles</name>
        <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.LifecyclesConfig">
          <field name="lifecycles">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.LifecyclesConfi
                  <field name="name"><string>lifecycle1</string></field>
                  <field name="publicationPlugin"><string>Authoring publication</string></field>
                  <field name="states">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.Lifecyc
                          <field name="state"><string>draft</string></field>
                          <field name="membership"><string>author:/platform/web-contributors</string></field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.Lifecyc
                          <field name="state"><string>pending</string></field>
                          <field name="membership"><string>author:/platform/web-contributors</string></field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.Lifecyc
                          <field name="state"><string>approved</string></field>
                          <field name="membership"><string>manager:/platform/web-contributors</string></field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.Lifecyc
                          <field name="state"><string>staged</string></field>
                          <field name="membership"><string>publisher:/platform/web-contributors</string></field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.wcm.extensions.publication.lifecycle.impl.Lifecyc
                          <field name="state"><string>published</string></field>
                          <field name="membership"><string>publisher:/platform/web-contributors</string></field>
                        </object>
                      </value>
                    </collection>
```

```
                </field>
              </object>
            </value>

          </collection>
        </field>
      </object>
    </object-param>
  </init-params>
 </component-plugin>
 <component-plugin>
  <name>AddContext</name>
  <set-method>addContext</set-method>
  <type>org.exoplatform.services.wcm.extensions.publication.context.ContextPlugin</type>
  <init-params>
    <object-param>
      <name>contexts</name>
      <object type="org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig">
        <field name="contexts">
          <collection type="java.util.ArrayList">
            <value>
              <object type="org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig$Cor
                <field name="name"><string>contextdefault</string></field>
                <field name="priority"><string>200</string></field>
                <field name="lifecycle"><string>lifecycle1</string></field>
              </object>
            </value>
          </collection>
        </field>
      </object>
    </object-param>
  </init-params>
 </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.wcm.extensions.publication.PublicationManager`
- **Name**: `AddLifecycle`
- **Set-method**: `addLifecycle`
- **Type**: `org.exoplatform.services.wcm.extensions.publication.lifecycle.StatesLifecyclePlugin`

3.2.2. AddNameSpacesPlugin

3.2.2.1. Fields

3.2.2.2. Sample configuration

```
<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>war:/conf/wcm-core/nodetypes/nodetypes-publication-config.xml</value>
              </values-param>
            </init-params>
          </component-plugin>
</external-component-plugins>
```

In which:

- **target component**: `org.exoplatform.services.jcr.RepositoryService`
- **name**: `add.namespaces`
- **set method**: `addPlugin`
- **type** `org.exoplatform.services.jcr.impl.AddNamespacesPlugin`

3.2.3. BaseActionPlugin

This is an abstract class and the parent of **ScriptActionPlugin**. You can create a link from this plugin to its children.

3.2.4. ContentTypeFilterPlugin

This is the configuration for the location of templates to inject in JCR.

```
 <external-component-plugins>
   <target-component>org.exoplatform.services.cms.templates.TemplateService</target-component>
   <component-plugin>
     <name>FilterContentTypeForWCMSpecificFolder</name>
     <set-method>addContentTypeFilterPlugin</set-method>
     <type>org.exoplatform.services.cms.templates.ContentTypeFilterPlugin</type>
     <description>this plugin is used to filter wcm nodetype</description>
     <init-params>
       <object-param>
         <name>cssFolderFilter</name>
         <description>only exo:cssFile can be created in exo:cssFolder</description>
         <object type="org.exoplatform.services.cms.templates.ContentTypeFilterPlugin$FolderFilterConfig">
           <field name="folderType">
             <string>exo:cssFolder</string>
           </field>
           <field name="contentTypes">
             <collection type="java.util.ArrayList">
               <value>
                 <string>exo:cssFile</string>
               </value>
             </collection>
           </field>
         </object>
       </object-param>
       <object-param>
         <name>jsFolderFolderFilter</name>
         <description>only exo:cssFile can be created in exo:cssFolder</description>
         <object type="org.exoplatform.services.cms.templates.ContentTypeFilterPlugin$FolderFilterConfig">
           <field name="folderType">
             <string>exo:jsFolder</string>
           </field>
           <field name="contentTypes">
             <collection type="java.util.ArrayList">
               <value>
                 <string>exo:jsFile</string>
               </value>
```

```
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.templates.TemplateService`
- **Name**: `addTemplates`
- **Set-method**: `addTemplates`
- **Type**: `org.exoplatform.services.cms.templates.impl.TemplatePlugin`

3.2.5. ContextPlugin

3.2.5.1. Fields

Objectype: *org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig*

Name	Type	Default Value	Description	
name	`string`	context2	The name of the context.	
priority	`string`	`100`	The context priority, the higher number indicates higher priority. Because a site may have several life cycles, the life cycle with higher priority will be excuted priorly.	
lifecycle	`string`	`lifecycle2`	The name of the life cycle.	
site	`string`	`acme`	The site that will apply the context configuration.	

3.2.5.2. Sample configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.wcm.extensions.publication.PublicationManager</target-component>
  <component-plugin>
    <name>AddContext</name>
    <set-method>addContext</set-method>
    <type>org.exoplatform.services.wcm.extensions.publication.context.ContextPlugin</type>
    <init-params>
```

```
      <object-param>
        <name>contexts</name>
        <object type="org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig">
          <field name="contexts">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.wcm.extensions.publication.context.impl.ContextConfig$Con
                  <field name="name"><string>context2</string></field>
                  <field name="priority"><string>100</string></field>
                  <field name="lifecycle"><string>lifecycle2</string></field>
                  <field name="site"><string>acme</string></field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.wcm.extensions.publication.PublicationManager`
- **Name**: `AddLifecycle`
- **Set-method**: `addLifecycle`
- **Type**: `org.exoplatform.services.wcm.extensions.publication.lifecycle.StatesLifecyclePlugin`

3.2.6. CreatePortalPlugin

This is an abstract class and the parent of **CreateTaxonomyPlugin** . You can create a link from this plugin to its children.

3.2.7. FriendlyPlugin

3.2.7.1. Fields

Object type: *org.exoplatform.services.wcm.friendly.impl.FriendlyConfig*

Name	Type	Default Value	Description
friendlyUrl	`string`	`documents`	The object that will be applied the friendly URL.
unfriendlyUrl	`string`	`/public/acme/detail?content-path=repository:collaboration`	The path to the location that will be applied the friendly URL.

3.2.7.2. Sample configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.wcm.friendly.FriendlyService</target-component>
  <component-plugin>
    <name>FriendlyService.addConfiguration</name>
    <set-method>addConfiguration</set-method>
    <type>org.exoplatform.services.wcm.friendly.impl.FriendlyPlugin</type>
```

```
    <description>Configures</description>
    <priority>100</priority>
    <init-params>
      <value-param>
        <name>enabled</name>
        <value>true</value>
      </value-param>
      <object-param>
        <name>friendlies.configuration</name>
        <object type="org.exoplatform.services.wcm.friendly.impl.FriendlyConfig">
          <field name="friendlies">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.wcm.friendly.impl.FriendlyConfig$Friendly">
                  <field name="friendlyUri"><string>documents</string></field>
                  <field name="unfriendlyUri"><string>/public/acme/detail?content-id=/repository/collaboration</
                </object>
              </value>
              <value>
                <object type="org.exoplatform.services.wcm.friendly.impl.FriendlyConfig$Friendly">
                  <field name="friendlyUri"><string>files</string></field>
                  <field name="unfriendlyUri"><string>/rest-ecmdemo/jcr/repository/collaboration</string></field
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.wcm.friendly.FriendlyService`
- **Name**: `FriendlyService.addConfiguration`
- **Set-method**: `addConfiguration`
- **Type**: `org.exoplatform.services.wcm.friendly.impl.FriendlyPlugin`

3.2.8. ImageThumbnailPlugin

3.2.8.1. Fields

This plugin will configure the file types of the image thumbnail.

Object type: *org.exoplatform.services.cms.thumbnail.impl.ThumbnailType*

Name	Type	Default Value	Description
mimeTypes	`Array list`	`java.util.ArrayList`	List of image thumbnail types

3.2.8.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cms.thumbnail.ThumbnailService</target-component>
          <component-plugin>
            <name>ImageThumbnailPlugin</name>
            <set-method>addPlugin</set-method>
            <type>org.exoplatform.services.cms.thumbnail.impl.ImageThumbnailPlugin</type>
```

```
            <init-params>
              <object-param>
                <name>thumbnailType</name>
                <description>Thumbnail types</description>
                <object type="org.exoplatform.services.cms.thumbnail.impl.ThumbnailType">
                                <field name="mimeTypes">
                                        <collection type="java.util.ArrayList">
                                                <value><string>image/jpeg</string></value>
                                                <value><string>image/png</string></value>
                                                <value><string>image/gif</string></value>
                                                <value><string>image/bmp</string></value>
                                                <value><string>image/tiff</string></value>
                                        </collection>
                                </field>
                </object>
              </object-param>
            </init-params>
          </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.thumbnail.ThumbnailService`
- **Name**: `ImageThumbnailPlugin`
- **Set-method**: `addPlugin`
- **Type**: `org.exoplatform.services.cms.thumbnail.impl.ImageThumbnailPlugin`

3.2.9. InitialWebcontentPlugin

As a portal is created, this plugin will deploy initial web-contents as the site artifact into the **Site Artifact** folder of that portal.

3.2.9.1. Fields

Object type: *org.exoplatform.services.deployment.DeploymentDescriptor$Target*

Name	Type	Default Value	Description
workspace	`string`	`collaboration`	The workspace where the initial web-contents will be deployed into.
nodePath	`string`	`/sites content/live//web contents/site artifacts`	The target node where the initial web-contents will be deployed into.
sourcePath	`string`	`war:/conf/sample-portal/acm/artifacts/resources/acm`	The path to the source that this plugin will get data.

3.2.9.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.wcm.portal.artifacts.CreatePortalArtifactsService</target-compone
```

```
    <component-plugin>
      <name>Initial webcontent artifact for each site</name>
      <set-method>addPlugin</set-method>
      <type>org.exoplatform.services.wcm.webcontent.InitialWebContentPlugin</type>
      <description>This plugin deploy some initial webcontent as site artifact to site artifact folder of portal
      <init-params>
        <object-param>
          <name>Portal logo data</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
            <field name="target">
              <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
                <field name="repository">
                  <string>repository</string>
                </field>
                <field name="workspace">
                  <string>collaboration</string>
                </field>
                <field name="nodePath">
                  <string>/sites content/live/{portalName}/web contents/site artifacts</string>
                </field>
              </object>
            </field>
            <field name="sourcePath">
              <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Logo.xml</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Portal signin data</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
            <field name="target">
              <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
                <field name="repository">
                  <string>repository</string>
                </field>
                <field name="workspace">
                  <string>collaboration</string>
                </field>
                <field name="nodePath">
                  <string>/sites content/live/{portalName}/web contents/site artifacts</string>
                </field>
              </object>
            </field>
            <field name="sourcePath">
              <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Signin.xml</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Portal searchbox data</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
            <field name="target">
              <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
                <field name="repository">
                  <string>repository</string>
                </field>
                <field name="workspace">
                  <string>collaboration</string>
                </field>
                <field name="nodePath">
                  <string>/sites content/live/{portalName}/web contents/site artifacts</string>
                </field>
              </object>
            </field>
            <field name="sourcePath">
              <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Searchbox.xml</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Portal navigation data</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
            <field name="target">
              <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
                <field name="repository">
```

```
              <string>repository</string>
            </field>
            <field name="workspace">
              <string>collaboration</string>
            </field>
            <field name="nodePath">
              <string>/sites content/live/{portalName}/web contents/site artifacts</string>
            </field>
          </object>
        </field>
        <field name="sourcePath">
          <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Navigation.xml</string
        </field>
      </object>
    </object-param>
    <object-param>
      <name>Portal footer data</name>
      <description>Deployment Descriptor</description>
      <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
        <field name="target">
          <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
            <field name="repository">
              <string>repository</string>
            </field>
            <field name="workspace">
              <string>collaboration</string>
            </field>
            <field name="nodePath">
              <string>/sites content/live/{portalName}/web contents/site artifacts</string>
            </field>
          </object>
        </field>
        <field name="sourcePath">
          <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Footer.xml</string>
        </field>
      </object>
    </object-param>
    <object-param>
      <name>Portal introduce data</name>
      <description>Deployment Descriptor</description>
      <object type="org.exoplatform.services.deployment.DeploymentDescriptor">
        <field name="target">
          <object type="org.exoplatform.services.deployment.DeploymentDescriptor$Target">
            <field name="repository">
              <string>repository</string>
            </field>
            <field name="workspace">
              <string>collaboration</string>
            </field>
            <field name="nodePath">
              <string>/sites content/live/{portalName}/web contents/site artifacts</string>
            </field>
          </object>
        </field>
        <field name="sourcePath">
          <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Introduce.xml</string>
        </field>
      </object>
    </object-param>
    <object-param>
      <name>Portal stylesheet 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/{portalName}/css</string>
            </field>
          </object>
        </field>
        <field name="sourcePath">
          <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/StylesheetGreen.xml</s
        </field>
```

```
          </object>
        </object-param>
        <object-param>
          <name>Portal images 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/{portalName}/medias/images</string>
                </field>
              </object>
            </field>
            <field name="sourcePath">
              <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme-templates/Images.xml</string>
            </field>
          </object>
        </object-param>
      </init-params>
    </component-plugin>
  </external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.wcm.portal.artifacts.CreatePortalArtifactsService`
- **Name**: `AddLifecycle`
- **Set-method**: `addPlugin`
- **Type**: `org.exoplatform.services.wcm.webcontent.InitialWebContentPlugin`

3.2.10. LinkDeploymentPlugin

3.2.10.1. Fields

Object type: *org.exoplatform.services.deployment.plugins.LinkDeploymentDescriptor*

Name	Type	Default Value	Description
sourcePath	`string`	`repository:collaboration:/sites content/live/acme/web contents/News/News1`	The path to the source where this plugin will get data.
targetPath	`string`	`repository:collaboration:/sites content/live/acme/categories/acme`	The path to the target where this plugin will deploy.

3.2.10.2. Sample 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.LinkDeploymentPlugin</type>
      <description>Link Deployment Plugin</description>
      <init-params>
        <object-param>
          <name>link01</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.plugins.LinkDeploymentDescriptor">
            <field name="sourcePath">
              <string>repository:collaboration:/sites content/live/acme/web contents/News/News1</string>
            </field>
            <field name="targetPath">
              <string>repository:collaboration:/sites content/live/acme/categories/acme</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>link02</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.plugins.LinkDeploymentDescriptor">
            <field name="sourcePath">
              <string>repository:collaboration:/sites content/live/acme/web contents/News/News2</string>
            </field>
            <field name="targetPath">
              <string>repository:collaboration:/sites content/live/acme/categories/acme</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>link03</name>
          <description>Deployment Descriptor</description>
          <object type="org.exoplatform.services.deployment.plugins.LinkDeploymentDescriptor">
            <field name="sourcePath">
              <string>repository:collaboration:/sites content/live/acme/web contents/News/News3</string>
            </field>
            <field name="targetPath">
              <string>repository:collaboration:/sites content/live/acme/categories/acme</string>
            </field>
          </object>
        </object-param>
      </init-params>
    </component-plugin>
  </external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.deployment.WCMContentInitializerService`
- **Name**: `Content Initializer Service`
- **Set-method**: `addPlugin`
- **Type**: `org.exoplatform.services.wcm.webcontent.InitialWebContentPlugin`

3.2.11. LockGroupsOrUsersPlugin

3.2.11.1. Fields

Object type *org.exoplatform.services.cms.lock.impl.LockGroupsOrUsersConfig*

Name	Type	Default Value	Description
settingLockList	`array list`	`java.util.ArrayList`	The list of the groups or user to be locked.

3.2.11.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cms.lock.LockService</target-component>
      <component-plugin>
        <name>predefinedLockGroupsOrUsersPlugin</name>
        <set-method>addLockGroupsOrUsersPlugin</set-method>
        <type>org.exoplatform.services.cms.lock.impl.LockGroupsOrUsersPlugin</type>
        <init-params>
          <object-param>
            <name>LockGroupsOrUsers.configuration</name>
            <description>configuration predefined groups or users for lock administrator</description>
            <object type="org.exoplatform.services.cms.lock.impl.LockGroupsOrUsersConfig">
              <field name="settingLockList">
                <collection type="java.util.ArrayList">
                                    <value><string>*:/platform/administrators</string></value>
                </collection>
              </field>
            </object>
          </object-param>
        </init-params>
      </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.lock.LockService`
- **Name**: `predefinedLockGroupsOrUsersPlugin`
- **Set-method**: `addLockGroupsOrUsersPlugin`
- **Type**: `org.exoplatform.services.cms.lock.impl.LockGroupsOrUsersPlugin`

3.2.12. ManageDrivePlugin

A drive can be considered as 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.

A drive is the combination of:

- Path: the root folder of the drive.
- View: how we can see contents, such as by list, thumbnails, coverflow.
- Role: the visibility to every users, a group or a single user.
- Options: allow you to specify whether to see hidden nodes or not and to create folders in this drive or not.

3.2.12.1. Fields

Object type: *org.exoplatform.services.cms.drives.**DriveData***

Field name	Type	Description
name	`String`	The name of drive which must be unique.

Field name	Type	Description
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. For example: `*:/platform/users`
icon	`String`	Url to the icon.
views	`String`	The list of views you want to use, separated by commas. For example: `simple-view,admin-view`
viewPreferences	`Boolean`	**User Preference** icon will be visible if true.
viewNonDocument	`Boolean`	Non-document types will be visible 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 node types that you can create as folders. For example: `nt:folder,nt:unstructured`.
allowNodeTypesOnTree	`String`	Allows you to filter node types in the navigation tree. For example, the default value is "*" to show all content types.

The following structure is used 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
                            </object-param>
                    </init-params>
            </component-plugin>
```

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

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

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

3.2.12.2. Sample 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>
```

3.2.13. ManageViewPlugin

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

3.2.13.1. Fields

- **target-component**: *org.exoplatform.services.cms.views.ManageViewService</target-component-*
- **Name**: `manage.view.plugin`
- **Set-method**: `setManageViewPlugin`
- **Type**: `org.exoplatform.services.cms.views.impl.ManageViewPlugin`
- **Description**: This plugin manages user views.

Object type: *org.exoplatform.services.cms.views.ViewConfig*

Name	Type	Default value	Description
name	`String`	`system-view`	The name of view which must be unique inside WCM.
permissions	`String`	*:/platform/administrators	Visibility of the view based on eXo rights.
template	`String`	`/exo:ecm/views/templates/ecm-explorer/SystemView`	Specify path to the template location.
tabList	`ArrayList`	`java.util.ArrayList`	Include a set of view names.

Object type: *org.exoplatform.services.cms.views.ViewConfig$Tab*

Name	Type	Default value	Description
tabName	`String`	`Info`	The name of tab which must be unique.
button	`String`	`viewNodeType; viewPermissions; viewProperties; showJCRStructure`	Specify a set of view component names.

Object type: *org.exoplatform.services.cms.views.TemplateConfig*

Name	Type	Default value	Description
type	`String`	`ecmExplorerTemplate`	Specify if a name is truly a class representing all properties of a view.
name	`String`	`system-view`	Specify a set of view component names.
warPath	`String`	`/ecm-explorer/SystemView.groovy`	Specify a template location to view.

3.2.13.2. 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</strin
                          <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; showJCRStruc
                                    </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>
```

3.2.14. PDFThumbnailPlugin

This plugin is to set the supported file types of PDF thumbnail. See also **ImageThumbnailPlugin** .

3.2.14.1. Fields

Object type: *org.exoplatform.services.cms.thumbnail.impl.ThumbnailType*

Name	Type	Default Value	Description
mimeTypes	`array list`	`java.util.ArrayList`	The MIME type of the pdf thumbnail.

3.2.14.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cms.thumbnail.ThumbnailService</target-component>
          <component-plugin>
            <name>ImageThumbnailPlugin</name>
            <set-method>addPlugin</set-method>
            <type>org.exoplatform.services.cms.thumbnail.impl.ImageThumbnailPlugin</type>
            <init-params>
              <object-param>
                <name>thumbnailType</name>
                <description>Thumbnail types</description>
                <object type="org.exoplatform.services.cms.thumbnail.impl.ThumbnailType">
                                <field name="mimeTypes">
                                        <collection type="java.util.ArrayList">
                                                <value><string>image/jpeg</string></value>
                                                <value><string>image/png</string></value>
                                                <value><string>image/gif</string></value>
                                                <value><string>image/bmp</string></value>
                                                <value><string>image/tiff</string></value>
                                        </collection>
                                </field>
                </object>
              </object-param>
            </init-params>
          </component-plugin>
    <component-plugin>
            <name>PDFThumbnailPlugin</name>
            <set-method>addPlugin</set-method>
            <type>org.exoplatform.services.cms.thumbnail.impl.PDFThumbnailPlugin</type>
            <init-params>
              <object-param>
                <name>thumbnailType</name>
                <description>Thumbnail types</description>
                <object type="org.exoplatform.services.cms.thumbnail.impl.ThumbnailType">
                                <field name="mimeTypes">
                                        <collection type="java.util.ArrayList">
                                                <value><string>application/pdf</string></value>
                                        </collection>
                                </field>
                </object>
              </object-param>
            </init-params>
          </component-plugin>
  </external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.thumbnail.ThumbnailService`
- **Name**: `ImageThumbnailPlugin`
- **Set-method**: `addPlugin`
- **Type**: `org.exoplatform.services.cms.thumbnail.impl.ImageThumbnailPlugin`

3.2.15. PorletTemplatePlugin

This plugin is used to import the view templates for Content List Viewer.

3.2.15.1. Fields

Object type: *org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig*

Name	Type	Default Value	Description
templateName	`string`	`UIContentListPresentationDefault.gtmpl`	The name of the GROOVY template.

3.2.15.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cms.views.ApplicationTemplateManagerService</target-component>
    <component-plugin>
      <name>clv.templates.plugin</name>
      <set-method>addPlugin</set-method>
      <type>org.exoplatform.services.cms.views.PortletTemplatePlugin</type>
      <description>This plugin is used to import views templates for Content List Viewer</description>
      <init-params>
        <value-param>
          <name>portletName</name>
          <value>content-list-viewer</value>
        </value-param>
        <value-param>
          <name>portlet.template.path</name>
          <value>war:/conf/wcm-artifacts/application-templates/content-list-viewer</value>
        </value-param>
        <object-param>
          <name>default.folder.list.viewer</name>
          <description>Default folder list viewer groovy template</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>UIContentListPresentationDefault.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>big.folder.list.viewer</name>
          <description>Default folder list viewer groovy template with Big Image</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>UIContentListPresentationBigImage.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>small.folder.list.viewer</name>
          <description>Default folder list viewer groovy template with Small Image</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>UIContentListPresentationSmall.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Two columns CLV template</name>
          <description>Two columns CLV template</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>TwoColumnsCLVTemplate.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
```

```
          <name>One column CLV template</name>
          <description>One column CLV template</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>OneColumnCLVTemplate.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Category Tree</name>
          <description>Category Tree</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>CategoryTree.gtmpl</string>
            </field>
            <field name="category">
              <string>navigation</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Category List</name>
          <description>Category List</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>CategoryList.gtmpl</string>
            </field>
            <field name="category">
              <string>navigation</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Tags Cloud</name>
          <description>Tags Cloud</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>TagsCloud.gtmpl</string>
            </field>
            <field name="category">
              <string>navigation</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Documents template</name>
          <description>Documents template</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>DocumentsTemplate.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>folder.list.viewer</name>
          <description>Default folder list viewer groovy template</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>ContentListViewerDefault.gtmpl</string>
            </field>
            <field name="category">
              <string>list</string>
            </field>
          </object>
        </object-param>
        <object-param>
          <name>Big hot news template</name>
          <description>Big hot news template</description>
          <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
            <field name="templateName">
              <string>BigHotNewsTemplateCLV.gtmpl</string>
            </field>
            <field name="category">
```

```
            <string>list</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>Events template</name>
        <description>Events template</description>
        <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
          <field name="templateName">
            <string>EventsTemplateCLV.gtmpl</string>
          </field>
          <field name="category">
            <string>list</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>default.paginator</name>
        <description>Default paginator groovy template</description>
        <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
          <field name="templateName">
            <string>UIPaginatorDefault.gtmpl</string>
          </field>
          <field name="category">
            <string>paginators</string>
          </field>
        </object>
      </object-param>
      <object-param>
        <name>empty.paginator</name>
        <description>Empty paginator groovy template</description>
        <object type="org.exoplatform.services.cms.views.PortletTemplatePlugin$PortletTemplateConfig">
          <field name="templateName">
            <string>UIPaginatorEmpty.gtmpl</string>
          </field>
          <field name="category">
            <string>paginators</string>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.views.ApplicationTemplateManagerService`
- **Name**: `clv.templates.plugin`
- **Set-method**: `addPlugin`
- **Type**: `org.exoplatform.services.wcm.webcontent.InitialWebContentPlugin`

3.2.16. PredefinedProcessesPlugin

3.2.16.1. Fields

Object type: org.exoplatform.services.workflow.ProcessesConfig

Name	Type	Default Value	Description
processLocation	`string`	`war:/conf/bp`	The path to the process.
predefinedProcess	`hash set`	`java.util.HashSet`	The list of the processes.

3.2.16.2. Sample configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.workflow.WorkflowServiceContainer</target-component>
   <component-plugin>
     <name>deploy.predefined.processes</name>
     <set-method>addPlugin</set-method>
     <type>org.exoplatform.services.workflow.PredefinedProcessesPlugin</type>
     <init-params>
       <object-param>
         <name>predefined.processes</name>
         <description>load of default business processes</description>
         <object type="org.exoplatform.services.workflow.ProcessesConfig">
           <field name="processLocation"><string>war:/conf/bp</string></field>
           <field name="predefinedProcess">
             <collection type="java.util.HashSet">
             <value><string>/exo-ecms-ext-workflow-bp-bonita-content-2.2.0-GA.jar</string></value>
                          <value><string>/exo-ecms-ext-workflow-bp-bonita-payraise-2.2.0-GA.jar</string></value>
             <value><string>/exo-ecms-ext-workflow-bp-bonita-holiday-2.2.0-GA.jar</string></value>
             </collection>
           </field>
         </object>
       </object-param>
     </init-params>
   </component-plugin>
 </external-component-plugins>
```

In which:

- **target component**: `org.exoplatform.services.workflow.WorkflowServiceContainer`
- **name**: `deploy.predefined.processes`
- **set-method**: `addPlugin`
- **type**: `org.exoplatform.services.workflow.PredefinedProcessesPlugin`

3.2.17. PublicationPlugin

This is an abstract class and the parent of **StageAndVersionPublicationPlugin** and **AuthoringPublicationPlugin** . You can create a link from this plugin to its childrens.

3.2.18. QueryPlugin

```
<external-component-plugins>
   <target-component>org.exoplatform.services.cms.queries.QueryService</target-component>
    <component-plugin>
      <name>query.plugin</name>
      <set-method>setQueryPlugin</set-method>
      <type>org.exoplatform.services.cms.queries.impl.QueryPlugin</type>
      <description>Nothing</description>
      <init-params>
              <value-param>
               <name>autoCreateInNewRepository</name>
               <value>true</value>
       </value-param>
       <value-param>
               <name>repository</name>
               <value>repository</value>
       </value-param>
       <object-param>
         <name>CreatedDocuments</name>
         <description>documents created by the current user</description>
         <object type="org.exoplatform.services.cms.queries.impl.QueryData">
```

```
          <field name="name"><string>Created Documents</string></field>
          <field name="language"><string>xpath</string></field>
          <field name="statement"><string>//*[(@jcr:primaryType = 'exo:article' or @jcr:primaryType = 'nt:file'
          <field name="permissions">
            <collection type="java.util.ArrayList">
              <value><string>*:/platform/users</string></value>
            </collection>
          </field>
          <field name="cachedResult"><boolean>false</boolean></field>
        </object>
      </object-param>
      <object-param>
        <name>CreatedDocumentsDayBefore</name>
        <description>documents created the day before</description>
        <object type="org.exoplatform.services.cms.queries.impl.QueryData">
          <field name="name"><string>CreatedDocumentDayBefore</string></field>
          <field name="language"><string>xpath</string></field>
          <field name="statement"><string>//element(*,exo:article)[@exo:dateCreated &lt; xs:dateTime('${Date}$'
          <field name="permissions">
            <collection type="java.util.ArrayList">
              <value><string>*:/platform/users</string></value>
            </collection>
          </field>
          <field name="cachedResult"><boolean>true</boolean></field>
        </object>
      </object-param>
      <object-param>
        <name>AllArticles</name>
        <description>All articles</description>
        <object type="org.exoplatform.services.cms.queries.impl.QueryData">
          <field name="name"><string>All Articles</string></field>
          <field name="language"><string>xpath</string></field>
          <field name="statement"><string>//element(*,exo:article) order by @exo:dateCreated descending</string
          <field name="permissions">
            <collection type="java.util.ArrayList">
              <value><string>*:/platform/users</string></value>
            </collection>
          </field>
          <field name="cachedResult"><boolean>true</boolean></field>
        </object>
      </object-param>
    </init-params>
      </component-plugin>
</external-component-plugins>
```

- **Target Component**: `org.exoplatform.services.cms.queries.QueryService`
- **Name**: `predefinedTaxonomyPlugin`
- **Set-method**: `setQueryPlugin`
- **Type**: `org.exoplatform.services.cms.queries.impl.QueryPlugin`

Value param	Possible value	Default value	Description
`autoCreateInNewRepository`	`boolean`	`true`	
`repository`	`string`	`repository`	repository to the target node.

Object param	Possible value	Default value	Description
`CreatedDocuments`	`string`	**`:/platform/users`**	documents created by the current user.
`AllArticles`	`string`	**`:/platform/users`**	All articles.
`CreatedDocumentsDayBefore`	`string`	**`:/platform/users`**	documents created on the

Object param	Possible value	Default value	Description
			day before.

3.2.19. RemovePortalPlugin

This is an abstract class and the parent of **RemoveTaxonomyPlugin**. You can create a link from this Plugin to its children.

3.2.20. ScriptActionPlugin

This is an abstract class and the parent of **RemoveTaxonomyPlugin**. You can create a link from this Plugin to its children.

3.2.21. ScriptPlugin

3.2.21.1. Fields

Object type: *org.exoplatform.services.cms.impl.ResourceConfig*

Name	Type	Default Value	Description	
resources	`array list`	`java.util.ArrayList`	The list of resources.	
name	`string`	-	The resource name.	

3.2.21.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cms.scripts.ScriptService</target-component>
            <component-plugin>
        <name>manage.script.plugin</name>
        <set-method>addScriptPlugin</set-method>
        <type>org.exoplatform.services.cms.scripts.impl.ScriptPlugin</type>
        <description>Nothing</description>
                    <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>predefinedScriptsLocation</name>
                        <value>war:/conf/dms-extension/dms/artifacts</value>
                </value-param>
                      <object-param>
                        <name>predefined.scripts</name>
                        <description>description</description>
                        <object type="org.exoplatform.services.cms.impl.ResourceConfig">
                          <field name="resources">
                            <collection type="java.util.ArrayList">
                              <value>
                                <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                                  <field name="name"><string>ecm-explorer/action/RSSScript.groovy</string></fiel
```

```
                    </object>
                  </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/SendMailScript.groovy</string><
                    </object>
                  </value>
          <value>
            <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
              <field name="name"><string>ecm-explorer/action/TrashFolderScript.groovy</string></field>
            </object>
          </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/EnableVersioningScript.groovy</
                    </object>
                  </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/AutoVersioningScript.groovy</st
                    </object>
                  </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/AddMetadataScript.groovy</strin
                    </object>
                  </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/TransformBinaryChildrenToTextSc
                    </object>
                  </value>
                  <value>
                                        <object type="org.exoplatform.services.cms.impl.Resource
                                          <field name="name"><string>ecm-explorer/action/G
                                        </object>
                  </value>
                  <value>
                                        <object type="org.exoplatform.services.cms.impl.Resource
                                          <field name="name"><string>ecm-explorer/action/P
                                        </object>
                  </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/PublishingRequestScript.groovy<
                    </object>
                  </value>
                  <value>
                    <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                      <field name="name"><string>ecm-explorer/action/AddTaxonomyActionScript.groovy<
                    </object>
                  </value>
                  <value>
            <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
              <field name="name"><string>ecm-explorer/widget/FillSelectBoxWithCalendarCategories.g
            </object>
          </value>
          <value>
            <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
              <field name="name"><string>ecm-explorer/widget/FillSelectBoxWithMetadatas.groovy</st
            </object>
          </value>
                <value>
                  <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                    <field name="name"><string>ecm-explorer/widget/FillSelectBoxWithWorkspaces.gro
                  </object>
                </value>
                <value>
                  <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                    <field name="name"><string>ecm-explorer/widget/FillSelectBoxWithNodeChildren.g
                  </object>
                </value>
                            <value>
                  <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                    <field name="name"><string>ecm-explorer/widget/FillSelectBoxWithLanguage.groov
                  </object>
                </value>
                <value>
                  <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
```

```
                                <field name="name"><string>ecm-explorer/interceptor/PreNodeSaveInterceptor.gro
                              </object>
                            </value>
                            <value>
                              <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                                <field name="name"><string>ecm-explorer/interceptor/PostNodeSaveInterceptor.gr
                              </object>
                            </value>
                            <value>
                              <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                                <field name="name"><string>ecm-explorer/interceptor/PostFilePlanInterceptor.gr
                              </object>
                            </value>
                            <value>
                              <object type="org.exoplatform.services.cms.impl.ResourceConfig$Resource">
                                <field name="name"><string>content-browser/GetDocuments.groovy</string></field
                              </object>
                            </value>
                          </collection>
                        </field>
                      </object>
                    </object-param>
                  </init-params>
                </component-plugin>
  </external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.scripts.ScriptService`
- **Name**: `manage.script.plugin`
- **Set-method**: `addScriptPlugin`
- **Type**: `org.exoplatform.services.cms.scripts.impl.ScriptPlugin`

3.2.22. StageAndVersionPublicationPlugin

3.2.22.1. Fields

object type: *org.exoplatform.services.cache.ExoCacheConfig*

Name	Type	Default Value	Description	
name	`string`	`wcm.compose`	The name of the cache which must be unique .	
maxSize	`string`	`200`	The max size of the cache object.	
liveTime	`string`	`600`	The time interval.	
implementation	`string`	`org.exoplatform.services.cache.concurrent.ConcurrentFIFOExoCache`	The algorithm that is used to hande the cache.	

3.2.22.2. Sample configuration

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

```
    <target-component>org.exoplatform.services.cache.CacheService</target-component>
    <component-plugin>
      <name>addExoCacheConfig</name>
      <set-method>addExoCacheConfig</set-method>
      <type>org.exoplatform.services.cache.ExoCacheConfigPlugin</type>
      <description>Configures the cache for query service</description>
      <init-params>
        <object-param>
          <name>cache.config.wcm.composer</name>
          <description>The default cache configuration</description>
          <object type="org.exoplatform.services.cache.ExoCacheConfig">
            <field name="name">
              <string>wcm.composer</string>
            </field>
            <field name="maxSize">
              <int>300</int>
            </field>
            <field name="liveTime">
              <long>600</long>
            </field>
            <field name="distributed">
              <boolean>false</boolean>
            </field>
            <field name="implementation">
              <string>org.exoplatform.services.cache.concurrent.ConcurrentFIFOExoCache</string>
            </field>
          </object>
        </object-param>
      </init-params>
    </component-plugin>
  </external-component-plugins>

  <external-component-plugins>
    <target-component>org.exoplatform.services.wcm.core.WebSchemaConfigService</target-component>
    <component-plugin>
      <name>StageAndVersionPublicationHandler</name>
      <set-method>addWebSchemaHandler</set-method>
      <type>org.exoplatform.services.wcm.publication.lifecycle.stageversion.StageAndVersionPublicationHandler</t
    </component-plugin>
  </external-component-plugins>

  <external-component-plugins>
    <target-component>org.exoplatform.services.listener.ListenerService</target-component>
    <component-plugin>
      <name>org.exoplatform.portal.config.DataStorage.pageCreated</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.page.CreatePageEventListener</type>
      <description>this listener update publication state of content when page is created</description>
    </component-plugin>
    <component-plugin>
      <name>org.exoplatform.portal.config.DataStorage.pageUpdated</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.page.UpdatePageEventListener</type>
      <description>this listener update publication state of content when page is updated</description>
    </component-plugin>
    <component-plugin>
      <name>org.exoplatform.portal.config.DataStorage.pageRemoved</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.page.RemovePageEventListener</type>
      <description>this listener update publication state of content when page is removed</description>
    </component-plugin>
    <component-plugin>
      <name>org.exoplatform.portal.config.DataStorage.navigationUpdated</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.navigation.UpdateNavigationEventListener</type>
      <description>this listener update publication state of content when navigation is updated</description>
    </component-plugin>
    <component-plugin>
      <name>org.exoplatform.portal.config.DataStorage.navigationRemoved</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.navigation.RemoveNavigationEventListener</type>
      <description>this listener update publication state of content when navigation is removed</description>
    </component-plugin>
  </external-component-plugins>

  <external-component-plugins>
    <target-component>org.exoplatform.services.listener.ListenerService</target-component>
    <component-plugin>
      <name>CmsService.event.postCreate</name>
```

```
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.post.PostCreateContentEventListener</type>
      <description>this listener will force the created document enroll to publication lifecycle</description>
    </component-plugin>
    <component-plugin>
      <name>CmsService.event.postEdit</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.post.PostEditContentEventListener</type>
      <description>this listener will change state of document to draft after editing</description>
    </component-plugin>
    <component-plugin>
      <name>FormGenerator.event.postCreate</name>
      <set-method>addListener</set-method>
      <type>org.exoplatform.services.wcm.publication.listener.post.PostCreateNodeTypeEventListener</type>
      <description>this listener will reinit the composer templates list when creating a new nodetype</descripti
    </component-plugin>
  </external-component-plugins>
```

In which:

- **target component**: `org.exoplatform.services.cache.CacheService`
- **name**: `addExoCacheConfig`
- **set method**: `addExoCacheConfig`
- **type** `org.exoplatform.services.cache.ExoCacheConfigPlugin`

.

- **target component**: `org.exoplatform.services.wcm.core.WebSchemaConfigService`
- **name**: `StageAndVersionPublicationHandler`
- **set method**: `addExoCacheConfig`
- **type** `org.exoplatform.services.cache.ExoCacheConfigPlugin`

.

- **target component**: `org.exoplatform.services.listener.ListenerService`
- **name**: `org.exoplatform.portal.config.DataStorage.pageCreated`
- **set method**: `addListener`
- **type** `org.exoplatform.services.wcm.publication.listener.page.CreatePageEventListener`

.

- **target component**: `org.exoplatform.services.listener.ListenerService`
- **name**: `org.exoplatform.portal.config.DataStorage.pageCreated`
- **set method**: `addListener`
- **type** `org.exoplatform.services.wcm.publication.listener.post.PostCreateContentEventListener`

3.2.23. TagPermissionPlugin

3.2.23.1. Fields

object type *org.exoplatform.services.cms.folksonomy.impl.TagPermissionConfig*

Name	Type	Default Value	Description
tagPermissionList	`array list`	**`:/platform/administrators`**	The users/groups that have the permission.

3.2.23.2. Sample configuration

```
<external-component-plugins>
  <target-component>org.exoplatform.services.cms.folksonomy.NewFolksonomyService</target-component>
    <component-plugin>
      <name>predefinedTagPermissionPlugin</name>
      <set-method>addTagPermissionPlugin</set-method>
      <type>org.exoplatform.services.cms.folksonomy.impl.TagPermissionPlugin</type>
      <init-params>
        <object-param>
          <name>TagPermission.configuration</name>
          <description>configuration predefined permission for tag to inject in jcr</description>
          <object type="org.exoplatform.services.cms.folksonomy.impl.TagPermissionConfig">
            <field name="tagPermissionList">
              <collection type="java.util.ArrayList">
                              <value><string>*:/platform/administrators</string></value>
              </collection>
            </field>
          </object>
        </object-param>
      </init-params>
    </component-plugin>
</external-component-plugins>
```

In which:

- **Target-component**: `org.exoplatform.services.cms.folksonomy.NewFolksonomyService`
- **Name**: `predefinedTagPermissionPlugin`
- **Set-method**: `addTagPermissionPlugin`
- **Type**: `org.exoplatform.services.cms.folksonomy.impl.TagPermissionPlugin`

3.2.24. TagStylePlugin

3.2.24.1. Fields

Object type: *org.exoplatform.services.cms.folksonomy.impl.TagStyleConfig*

Name	Type	Default Value	Description
autoCreatedInNewRepository	`boolean`	`true`	Specify whether the tag style is added automactically in new

Name	Type	Default Value	Description
			repositstory or not.
repository	`string`	`repository`	Name of the repository where the tag style is added.
tagStyleList	`array list`	`Array list`	The list of tag styles.

Object type: *org.exoplatform.services.cms.folksonomy.impl.TagStyleConfig$HtmlTagStyle*

Name	Type	Default Value	Description
name	`string`	-	The name of the tag.
tagRate	`string`	-	The number of times that a tag is used which will decide the respective tag style.
htmlStyle	`string`	-	The HTML code that defines the style.

3.2.24.2. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cms.folksonomy.NewFolksonomyService</target-component>
            <component-plugin>
                        <name>predefinedTagStylePlugin</name>
                  <set-method>addTagStylePlugin</set-method>
                  <type>org.exoplatform.services.cms.folksonomy.impl.TagStylePlugin</type>
                  <init-params>
                          <object-param>
                                  <name>htmStyleForTag.configuration</name>
                <description>configuration predefined html style for tag to inject in jcr</description>
                <object type="org.exoplatform.services.cms.folksonomy.impl.TagStyleConfig">
                        <field name="autoCreatedInNewRepository"><boolean>true</boolean></field>
                        <field name="repository"><string>repository</string></field>
                        <field name="tagStyleList">
                                <collection type="java.util.ArrayList">
                                        <value>
                                                <object type="org.exoplatform.services.cms.folksonomy.impl.Tag
                                                        <field name="name"><string>normal</string></field>
                                                    <field name="tagRate"><string>0..2</string></field>
                                                    <field name="htmlStyle">
                                                          <string>font-size: 12px; font-weight: bold; color: #
                                                    </field>
                                                  <field name="description"><string>Normal style for tag</stri
                                                </object>
                                        </value>
                                        <value>
                                                <object type="org.exoplatform.services.cms.folksonomy.impl.Tag
                                                  <field name="name"><string>interesting</string></field>
                                                  <field name="tagRate"><string>2..5</string></field>
                                                  <field name="htmlStyle">
                                                          <string>font-size: 13px; font-weight: bold; color: #
                                                  </field>
                                                  <field name="description"><string>Interesting style for tag<
                                                </object>
                                        </value>
                                        <value>
                                                <object type="org.exoplatform.services.cms.folksonomy.impl.Tag
                                                  <field name="name"><string>attractive</string></field>
                                                  <field name="tagRate"><string>5..7</string></field>
```

```
                                                <field name="htmlStyle">
                                                        <string>font-size: 15px; font-weight: bold; color: b
                                                </field>
                                                <field name="description"><string>attractive style for tag</
                                        </object>
                                </value>
                                <value>
                                        <object type="org.exoplatform.services.cms.folksonomy.impl.Tag
                                                <field name="name"><string>hot</string></field>
                                                <field name="tagRate"><string>7..10</string></field>
                                                <field name="htmlStyle">
                                                        <string>font-size: 18px; font-weight: bold; color: #
                                                </field>
                                                <field name="description"><string>hot style for tag</string>
                                        </object>
                                </value>
                                <value>
                                        <object type="org.exoplatform.services.cms.folksonomy.impl.Tag
                                                <field name="name"><string>hottest</string></field>
                                                <field name="tagRate"><string>10..*</string></field>
                                                <field name="htmlStyle">
                                                        <string>font-size: 20px; font-weight: bold; color: r
                                                </field>
                                                <field name="description"><string>hottest style for tag</str
                                        </object>
                                </value>
                        </collection>
                </field>
        </object>
                </object-param>
          </init-params>
        </component-plugin>
</external-component-plugins>
```

- **target component**: `org.exoplatform.services.cache.CacheService`
- **name**: `addExoCacheConfig`
- **set-method**: `addExoCacheConfig`
- **type**: `org.exoplatform.services.cache.ExoCacheConfigPlugin`

3.2.25. TaxonomyPlugin

3.2.25.1. Fields

Target Component: `org.exoplatform.services.cms.taxonomy.TaxonomyService`

Name: `predefinedTaxonomyPlugin`

Set-method: `addTaxonomyPlugin`

Type: *org.exoplatform.services.cms.taxonomy.impl.TaxonomyPlugin*

Value param	Type	Default value	Description
autoCreateInNewRepository	`Boolean`	`true`	Enable/Disable the creation of the taxonomies in the newly created repository.
repository	`String`	`repository`	Name of the repository where taxonomies are created.

Value param	Type	Default value	Description
workspace	`String`	`dms-system`	Name of the workspace where taxonomies are created.
treeName	`String`	`system`	Name of the taxonomy tree to be created.

Object param	Type	Default value	Description
permission.configuration	`String`	`org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfi`	Access permission for the whole taxonomy tree.
predefined.actions	`string`	`org.exoplatform.services.cms.actions.impl.ActionConfig`	Predefined actions for the taxonomy tree root node.
taxonomy.configuration	`String`	`org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfi`	Access permissions for each taxonomy in tree.

Object type: *org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig*

Name	Type	Description
taxonomies	`java.util.ArrayList<TaxonomyConfig$Taxonomy>`	List of taxonomies to be configured with permission.

Object type: *org.exoplatform.services.cms.taxonomy.impl.TaxonomyConfig$Taxonomy*

Name	Type	Default value	Description
name	`String`	`name`	Name of the taxonomy.
path	`String`	`cms`	Path to the taxonomy in the taxonomy tree.
permissions	`Arraylist`	`java.util.ArrayList<TaxonomyTreePermissionUserPermission$Pe`	List of permissions for users or groups to access the taxonomy.

Object type: *org.exoplatform.services.cms.actions.impl.ActionConfig*

Name	Type	Default value	Description
actions	`Arraylist`	`java.util.ArrayList<ActionConfig$TaxonomyAction>`	List of actions created for the taxonomy tree root node.

Object type: *org.exoplatform.services.cms.actions.impl.ActionConfig$TaxonomyAction*

Field name	Type	Default value	Description
type	`String`	`exo:taxonomyAction`	Type of the action.
name	`String`	`taxonomyAction`	Name of the action.
description	`String`	`N/A`	Description of the action.
homePath	`String`	`dms-system:/exo:ecm/exo:taxonomyTrees/exo:actionStorage/System`	Location of node where the action node is stored.
targetWspace	`String`	`N/A`	When a new node is created in the node whose path is defined in the **homePath** field, it will be moved to a new location automatically. The target workspace is specified in this field.
targetPath	`String`	`collaboration`	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	`Arraylist`	`java.util.ArrayList<String>`	The life cycle phases, in which the action is activated.
roles	`String`	*:/platform/administrators	The users or groups only with whom the action is activated.
mixins	`Arraylist`	`java.util.ArrayList<ActionConfig$Mixin>`	List of mixin types that are affected by this action.

Object type: *org.exoplatform.services.cms.actions.impl.ActionConfig$Mixin*

Field name	Type	Default value	Description
name	`String`	`mix:affectedNodeTypes`	Mixin node type name that will be added. The node type is used to store the affected node types.
properties	`String`	`exo:affectedNodeTypeNames,exo:addedNodeTypeNames,exo:removedNodeTypeNames,exo:broadcast,exo:sa`	The name of the properties that store all the node types affected by the action.

3.2.25.2. Sample configuration

```
<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.c
                                                                        <field name="permissions">
                                                                                <collection type="java.u
                                                                                        <value>
                                                                                                <object
                                                                                                        </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.c
                                                                        <field name="type"><string>exo:t
                                                                        <field name="name"><string>taxon
                                                                        <field name="description"><strin
                                                                        <field name="homePath"><string>c
                                                                        <field name="targetWspace"><stri
                                                                        <field name="targetPath"><string
                                                                        <field name="lifecyclePhase">
                                                                                <collection type="java.u
                                                                                        <value><string>r
                                                                                </collection>
                                                                        </field>
                                                                        <field name="roles"><string>*:/p
                                                                        <field name="mixins">
                                                                                <collection type="java.u
                                                                                        <value>
                                                                                                <object
```

```
                                                                                    </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</descr
                    <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.c
                                                            <field name="name"><string>cmsTa
                                                            <field name="path"><string>/cms<
                                                            <field name="permissions">
                                                                    <collection type="java.u
                                                                            <value>
                                                                                    <object




                                                                                    </object
                                                                            </value>
                                                                    </collection>
                                                            </field>
                                                    </object>
                                            </value>

                                    </collection>
                            </field>
                    </object>
            </object-param>
        </init-params>
    </component-plugin>
</external-component-plugins>
```

3.2.26. TemplatePlugin

A template is a presentation to display the saved information.

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

The node type template is used to edit and display the node content. Each node type has one *dialog1.gtmpl* file (dialog template) for editing/creating a node and one *view1.gtmpl* file (view template) for viewing the node content. Using the dialog template, you can specify a dialog whose fields correspond to the properties of the node you want to edit their values. When this template is rendered, each specified field will appear with a data input box for you to edit. Note that you do not have to design a dialog in which all data of the node are listed to be edited. You can just list the subset of node data you want to edit. Like the dialog template, the view template renders information of the node. You just need to create the template and specify which data fields to be displayed. With this kind of template, node information is only displayed but can not be edited. See details atContentType.

3.2.26.1. Fields

3.2.26.1.1. Init-params

Configuration name	Data type	Default value	Description
autoCreateInNewRepository	`Boolean`	`true`	Enable the application to import predefined templates at the start-up of template service automatically.
storedLocation	`String`	`war:/conf/dms-extension-content-ifaces/templates`	Location of stored templates.
repository	`String`	`repository`	Location of stored templates.
object-param	`Structure`	`N/A`	Configuration for each instance.

3.2.26.1.2. Object-param

Configuration name	Data type	Default Value	Description
name	`String`	`template.configuration`	The name of this configuration.
description	`String`	`configuration for the location of templates to inject in jcr`	Description of template instance.
object-type	`Structure`	`N/A`	Detailed configuration for each instance type.

Object-type defines all available template files, using the "collection type" configuration**.**

3.2.26.1.3. Object

type: It is the name of each object type. It means the type of template, the further configurations for this type are defined by some specified fields.

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 display of the title for this node.

There are three further information related to the presentation of each template:

- **`referencedView`** : Determine how to display a view.

- **`referencedDialog`** : Determine how to display a dialog to input information.

- **`referencedSkin`** : Determine the stylesheet for displaying.

Each type includes some definitions as the `object type="org.exoplatform.services.cms.templates.impl.TemplateConfig$Template"` with the fields as the properties.

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

3.2.26.2. Sample configuration

This below 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 node type.*

```
<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></fiel
                          <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>
```

3.2.27. WCMPublicationPlugin

3.2.27.1. Component configuration

Configuration name	Data type	Sample value	Description
key	`String`	`org.exoplatform.services.cms.publication.WCMPublicationS`	Service class location.
type	`String`	`org.exoplatform.services.cms.publication.WCMPublicationS`	Service implementation location.
component-plugins	`Object`	N/A	The plugins that are used inside the component.
init-params	`Object`	N/A	The initial parameter that will be used inside the component.

3.2.27.1.1. Component-plugin configuration

- **Description**:

Configuration name	Data type	Sample value	Description
name	`String`	`Simple publication`	Name of component-plugin.
set-method	`String`	`addPublicationPlugin`	Set method to start using the plugin.
type	`String`	`org.exoplatform.services.wcm.publication.lifecycle.simpl`	Class path location of plugin.

3.2.27.1.2. Init-params configuration

The parameter will be set in pair: name-value and will be placed inside the <value-param> object.

Configuration name	Data type	Sample value
name	`String`	`useCache`
value	`String`	`true`

Also, the parameter can be set in an object, with further information which will be the object's properties.

Configuration name	Data type	Sample value	Description
name	`String`	`cache.config.wcm.composer`	The name of object.
description	`String`	N/A	The brief description about the object.
object	`Object`	N/A	This object will be passed to component as a parameter, this object includes fieldset properties.

3.2.27.2. external-component-plugins configuration

Configuration name	Data type	Sample value	Description
target-component	`String`	`org.exoplatform.services.cache.CacheConfigService`	External plugin's class location.
component-plugin	`Object`	N/A	Configuration of component that will be used as a plugin.

3.2.27.3. Example

3.2.27.3.1. Component that use plugins example

Example of component using plugins.

```
<component>
    <key>org.exoplatform.services.wcm.publication.WCMPublicationService</key>
    <type>org.exoplatform.services.wcm.publication.WCMPublicationServiceImpl</type>
    <component-plugins>
      <component-plugin>
        <name>States and versions based publication</name>
        <set-method>addPublicationPlugin</set-method>
        <type>org.exoplatform.services.wcm.publication.lifecycle.stageversion.StageAndVersionPublicationPlugin</
        <description>This publication lifecycle publish a web content or DMS document to a portal page with more
      </component-plugin>
      <component-plugin>
        <name>Simple publication</name>
        <set-method>addPublicationPlugin</set-method>
        <type>org.exoplatform.services.wcm.publication.lifecycle.simple.SimplePublicationPlugin</type>
        <description>This publication lifecycle publish a web content or DMS document to a portal page without v
      </component-plugin>
    </component-plugins>
</component>
```

3.2.27.3.2. Component with init-params example

```
<component>
    <key>org.exoplatform.services.wcm.publication.WCMComposer</key>
    <type>org.exoplatform.services.wcm.publication.WCMComposerImpl</type>
    <init-params>
      <value-param>
        <name>useCache</name>
        <value>true</value>
      </value-param>
    </init-params>
</component>
```

3.2.27.3.3. Sample configuration

```
<external-component-plugins>
    <target-component>org.exoplatform.services.cache.CacheService</target-component>
    <component-plugin>
      <name>addExoCacheConfig</name>
      <set-method>addExoCacheConfig</set-method>
      <type>org.exoplatform.services.cache.ExoCacheConfigPlugin</type>
      <description>Configures the cache for query service</description>
      <init-params>
        <object-param>
          <name>cache.config.wcm.composer</name>
          <description>The default cache configuration</description>
          <object type="org.exoplatform.services.cache.ExoCacheConfig">
            <field name="name">
              <string>wcm.composer</string>
            </field>
            <field name="maxSize">
              <int>300</int>
            </field>
            <field name="liveTime">
              <long>600</long>
            </field>
            <field name="distributed">
              <boolean>false</boolean>
            </field>
            <field name="implementation">
              <string>org.exoplatform.services.cache.concurrent.ConcurrentFIFOExoCache</string>
            </field>
          </object>
        </object-param>
      </init-params>
    </component-plugin>
</external-component-plugins>
```

3.2.28. XMLdeploymentPlugin

When a site is created, most of end-users want to see something in the page instead of a blank page, so you need this service for deploying some "default" contents, such as Banner, Footer, Navigation, Breadcrumb.

There are two main cases to use:

- The site is created only one time when the database is cleaned.
- The site is created at runtime, when a user uses the core features of the GateIn portal.

3.2.28.1. Fields

init-params

Element	Type	Description
object-param	`Object`	The parameter which is an Object.

object-param

Element	Type	Default value	object-param
name	`String`	`ACME Logo data`	The name of this object parameter.
description	`String`	`Deployment Descriptor`	The description of this object parameter.
object	`Class`	`org.exoplatform.services.deployment.DeploymentDescriptor`	The object of this object parameter.

object

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

org.exoplatform.services.deployment.DeploymentDescriptor

Name	Type	Default value	Description
target	`Object`	`org.exoplatform.services.deployment.DeploymentDescriptor (*)`	The target node which will contain the imported node.
sourcePath	`String`	`war:/conf/sample-portal/acme/acme-resources/acm`	The absolute path of the XML file.
cleanupPublication	`Boolean`	`false`	Decide when the

Name	Type	Default value	Description
			publication lifecycle is cleaned up in the target folder after importing the data. true: allow false:not allow
versionHistoryPath	`String`	`war:/conf/sample-portal/com/acme/resources/acm`	The absolute path of the version history file.

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

Field	Type	Default value	Description
repository	`String`	`repository`	The repository of the target node.
repository	`String`	`repository`	The repository of the target node.
workspace	`String`	`collaboration`	The workspace of the target node.
nodePath	`String`	`/sites content/live/acme/web contents/site artifacts`	The path of the target node.

3.2.28.2. Sample 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>
          <field name="versionHistoryPath">
            <string>war:/conf/sample-portal/wcm/artifacts/site-resources/acme/Logo_versionHistory.zip</string>
          </field>
          <field name="cleanupPublication">
            <boolean>true</boolean>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

Chapter 4. Developer references

4.1. WCM Templates

4.1.1. Content types

Overview

The templates are applied to a node type or a metadata mixin type. There are two types of templates:

- **Dialogs**: are html forms that allow to create node instances.
- **Views**: are html fragments used to display nodes.

From the ECM admin portlet, the *Manage Template* lists existing node types 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).*

Document Type

1. The checkbox defines if the node type should be considered as the **Document type** or not. Content Explorer considers such nodes as user content and applies the following behavior:

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

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

4.1.1.1. 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 powerful syntax.

4.1.1.1.1. Common parameters

These following parameters are common and can be used for all input fields.

Parameter	Type	Required	Description	Example
jcrPath	`String`		Relative path inside the current node	`jcrPath=/node/exo:title`
mixintype	`String` with comma		List of mixin types	

Parameter	Type	Required	Description	Example
	, character		you want to initialize when creating the content.	mixintype=mix:i18n mixintype=mix:votable,mix:co
validate	String with comma , character		List of validators you want to apply to the input. Possible values are: name, email, number, empty, null, datetime, length OR validator classes	validate=empty validate=empty,name validate=org.exoplatform.web
editable	String		The input will be editable only if the value of this parameter is if-null and the value of this input is null or blank.	editable=if-null
multiValues	Boolean		Show a multi-valued component if true and must be used only with corresponding multi-valued properties. The default value of this parameter is false	multiValues=true
visible	Boolean		The input is visible if this value is true	visible=true

See also:

- Text field
- Hidden field
- Text area field

- Rich text field
- Calendar field
- Upload field
- Radio field
- Select box field
- Checkbox field
- Mixin field
- Action field

Note

`mixintype` can be used only in the root node field (commonly known as the name field).

4.1.1.1.2. Text Field

4.1.1.1.2.1. Additional parameters

See also: Common parameters

4.1.1.1.2.2. Example

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

4.1.1.1.3. Hidden Field

4.1.1.1.3.1. Additional parameters

See also: Common parameters

4.1.1.1.3.2. Example

```
String[] hiddenField5 = ["jcrPath=/node/jcr:content/dc:date", "visible=false"];
  uicomponent.addCalendarField("hiddenInput5", hiddenField5);
```

4.1.1.1.4. Text Area Field

4.1.1.1.4.1. Additional parameters

Parameter	Type	Required	Description	Example
rows	`Number`		The initial text area's number of rows. The value is `10` by default.	`rows=20`

Parameter	Type	Required	Description	Example
cols	`Number`		The initial text area's number of cols. The value is `30` by default .	`cols=50`

See also: Common parameters

4.1.1.1.4.2. Example

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

4.1.1.1.5. Rich Text Field

4.1.1.1.5.1. Additional parameters

Parameter	Type	Required	Description	Example
options	`String` with semicolon `;` character		Some options for CKEditor field: toolbar, width and height.	`options=CompleteWCM;width:'1`

Parameter	Type	Required	Description	Example
toolbar	`String`		The predefined toolbar for CKEditor. The value can be: `Default`, `Basic`, `CompleteWCM`, `BasicWCM`, `SuperBasicWCM`	`options=CompleteWCM.`
width	`String`		The width of CKEditor. Its value can be the percent of pixel.	`options=width:'100%'`
height	`String`		The height of CKEditor. Its value can be the percent of pixel.	`options=height:'200px'`

See also: Common parameters

4.1.1.1.5.2. Example

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

4.1.1.1.6. Calendar Field

4.1.1.1.6.1. Additional parameters

Parameter	Type	Required	Description	Example
options	`String`		An option for the calendar field: Display time	`options=displaytime`

See also: Common parameters

4.1.1.1.6.2. Example

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

4.1.1.1.7. Upload Field

4.1.1.1.7.1. Additional parameters

See also: Common parameters

4.1.1.1.7.2. Example

When you create an upload form, there are two main ways to store an

- If you want to store as property, do as follows:

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

- If you want to store as node, do as follows:

```
<%
  String[] hiddenField1 = ["jcrPath=/node/exo:image", "nodetype=nt:resource", "visible=false"] ;
  String[] hiddenField2 = ["jcrPath=/node/exo:image/jcr:encoding", "visible=false", "UTF-8"] ;
  String[] hiddenField3 = ["jcrPath=/node/exo:image/jcr:lastModified", "visible=false"] ;
  uicomponent.addHiddenField("hiddenInput1", hiddenField1) ;
  uicomponent.addHiddenField("hiddenInput2", hiddenField2) ;
  uicomponent.addHiddenField("hiddenInput3", hiddenField3) ;

  String[] fieldMedia = ["jcrPath=/node/exo:image"] ;
  uicomponent.addUploadField("media", fieldMedia) ;
%>
```

- But, this code is not complete. If you want to display the **upload** field, the image must be blank, otherwise you can display the image and an action enables you to remove it. You can do as follows:

```
<%
        def image = "image";
        // If you're trying to edit the document
        if(uicomponent.isEditing()) {
                def curNode = uicomponent.getNode();
                // If the image existed
                if (curNode.hasNode("exo:image")) {
                        def imageNode = curNode.getNode("exo:image") ;
                        // If the image existed and available
                        if (imageNode.getProperty("jcr:data").getStream().available() > 0 && (uicomponent.findCo
                                def imgSrc = uicomponent.getImage(curNode, "exo:image");
                                def actionLink = uicomponent.event("RemoveData", "/exo:image");
                                %>
                                        <div>
                                                <img src="$imgSrc" width="100px" height="80px"/>
                                                <a href="$actionLink">
                                                        <img src="/eXoResources/skin/DefaultSkin/background/Blan
                                                </a>
                                        </div>
                                <%
                        } else {
                                String[] fieldImage = ["jcrPath=/node/exo:image/jcr:data"] ;
                                uicomponent.addUploadField(image, fieldImage) ;
                        }
                } else {
                                String[] fieldImage = ["jcrPath=/node/exo:image/jcr:data"] ;
                                uicomponent.addUploadField(image, fieldImage) ;
                }
        }  else if(uicomponent.dataRemoved()) {
                String[] fieldImage = ["jcrPath=/node/exo:image/jcr:data"] ;
                uicomponent.addUploadField(image, fieldImage) ;
        } else {
                String[] fieldImage = ["jcrPath=/node/exo:image/jcr:data"] ;
                uicomponent.addUploadField(image, fieldImage) ;
        }
%>
```

4.1.1.1.8. Radio Field

4.1.1.1.8.1. Additional parameters

Parameter	Type	Required	Description	Example
options	`String` with comma `,` characters		Some radio values	`options=radio1,radio2,radio3`

See also: Common parameters

4.1.1.1.8.2. Example

```
<%
  String[] fieldDeep = ["jcrPath=/node/exo:isDeep", "defaultValues=true", "options=radio1,radio2,radio3"];
  uicomponent.addRadioBoxField("isDeep", fieldDeep);
%>
```

4.1.1.1.9. Select box Field

Parameter	Type	Required	Description	Example
options	`String` with comma `,` characters		Some option values	`options=option1,option2,opti`

See also: Common parameters

4.1.1.1.9.1. Example

```
<%
  String[] fieldDeep = ["jcrPath=/node/exo:isDeep", "defaultValues=true", "options=checkbox1,checkbox2,checkbox3
  uicomponent.addCheckBoxField("isDeep", fieldDeep);
%>
```

4.1.1.1.10. Checkbox Field

4.1.1.1.10.1. Additional parameters

Parameter	Type	Required	Description	Example
options	`String` with comma `,` characters		Some checkbox values	`options=checkbox1,checkbox2,`

See also: Common parameters

4.1.1.1.10.2. Example

```
<%
  String[] fieldDeep = ["jcrPath=/node/exo:isDeep", "defaultValues=true", "options=checkbox1,checkbox2,checkbox3
  uicomponent.addCheckBoxField("isDeep", fieldDeep);
%>
```

4.1.1.1.11. Mixin Field

4.1.1.1.11.1. Additional parameters

See also: Common parameters

4.1.1.1.11.2. Example

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

4.1.1.1.12. Action Field

4.1.1.1.12.1. Additional parameters

Parameter	Type	Required	Description	Example
selectorClass	`String`		The component to display.	`selectorClass=org.exoplatfor`
selectorIcon	`String`		The action icon	`selectorIcon=SelectPath24x24`

Depending on the `selectorClass`, some other parameters can be added.

For example: The component `org.exoplatform.ecm.webui.tree.selectone.UIOneNodePathSelector` needs the following parameters:

Parameter	Type	Required	Description	Example
workspaceField	`String`		The field which enables you to select a workspace.	`workspaceField=targetWorkspa`

The component `org.exoplatform.ecm.webui.selector.UIPermissionSelector` does not need any special parameters.

See also: Common parameters

4.1.1.1.12.2. Example

```
<%
  String[] fieldPath = ["jcrPath=/node/exo:targetPath", "selectorClass=org.exoplatform.ecm.webui.tree.selectone.
  uicomponent.addActionField("targetPath", fieldPath)  ;
%>
```

4.1.1.1.13. Interceptors

To add an interceptor to a dialog, you 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 interceptor: `prev` or `post`.

4.1.1.1.13.1. Example

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

4.1.1.1.14. How to add new ECM template with tabs

To avoid refreshing the first tab for every action execution, add a new private function to the template with tabs. In the template, you must insert a new piece of code like the following:

```
private String getDisplayTab(String selectedTab) {
  if ((uicomponent.getSelectedTab() == null && selectedTab.equals("mainWebcontent"))
                        || selectedTab.equals(uicomponent.getSelectedTab())) {
                return "display:block";
        }
        return "display:none";
}

private String getSelectedTab(String selectedTab) {
        if (getDisplayTab(selectedTab).equals("display:block")) {
                return "SelectedTab";
        }
        return "NormalTab";
}
```

Changing in every event of onclick must be done like the following:

```
<div class="UITab NormalTabStyle">
  <div class="<%=getSelectedTab("mainWebcontent")%>
    ">
    <div class="LeftTab">
      <div class="RightTab">
        <div class="MiddleTab" onClick="<%=uicomponent.event("ChangeTab", "mainWebcontent")%>"><%=_ctx.appRes("W
      </div>
    </div>
  </div>
</div>

<div class="UITab NormalTabStyle">
  <div class="<%=getSelectedTab("illustrationWebcontent")%>
    ">
    <div class="LeftTab">
      <div class="RightTab">
        <div class="MiddleTab" onClick="<%=uicomponent.event("ChangeTab", "illustrationWebcontent")%>"><%=_ctx.a
      </div>
    </div>
  </div>
</div>

<div class="UITab NormalTabStyle">
  <div class="<%= getSelectedTab("contentCSSWebcontent")%>
    ">
    <div class="LeftTab">
      <div class="RightTab">
        <div class="MiddleTab" onClick="<%=uicomponent.event("ChangeTab", "contentCSSWebcontent")%>"><%=_ctx.app
      </div>
    </div>
  </div>
</div>
```

Finally, to display the selected tab, simply add it to the style of UITabContent class.

```
<div class="UITabContent" style="<%=getDisplayTab("mainWebcontent")%>">
```

4.1.1.2. 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")%>
```

4.1.2. List of Contents

4.1.2.1. Content List Template

Content List Template allows you to view the content list with various templates. eXo Platform supports the following content list templates:

Template	Description
BigHotNewsTemplateCLV.gtmpl	Allows the contents to display under one column with a content list. The illustration image of each content is displayed above the content.
ContentListViewerDefault.gtmpl	Its function is similar to **BigHotNewsTemplateCLV.gtmpl**. The illustration image of each content is bigger.
DocumentsTemplate.gtmpl	Allows the contents to display under a content list with a NodeType icon or the illustration image on the left of the corresponding content.
EventsTemplateCLV.gtmpl	Its function is similar to **BigHotNewsTemplateCLV.gtmpl**, but the illustration image of each content is smaller.
OneColumnCLVTemplate.gtmpl	Allows the contents to display under one column. The illustration image of each content is displayed on its left.

Template	Description
TwoColumnsCLVTemplate.gtmpl	Allows the contents to display under two columns. The illustration image of each content is displayed on its left.
UIContentListPresentationBigImage.gtmpl	Its function is similar to **BigHotNewsTemplateCLV.gtmpl**, but the illustration image of each content is bigger than the image displayed with **ContentListViewerDefault.gtmpl** and the text font is different.
UIContentListPresentationDefault.gtmpl	Its function is similar to **BigHotNewsTemplateCLV.gtmpl**, but the illustration image of each content is smaller and the text font is different.
UIContentListPresentationSmall.gtmpl	Allows the contents to display under one column with a content list. The images are displayed on the left of the corresponding content and smaller than the images of the other templates.

4.1.2.2. Category Navigation Template

Category Navigation Template displays all contents under the categories.

Template	Description
CategoryList.gtmpl	Displays the categories as a navigation bar.
CategoryTree.gtmpl	Displays the categories as a tree.
TagsCloud.gtmpl	Displays all tags of the contents.

4.2. WCM Explorer

4.2.1. CSS

- By using WCM, all the stylesheets of each site can be managed online easily. You do not need to access the file system to modify and wait until the server has been restarted. For the structure, each site has its own CSS folder which can contain one or more CSS files. These CSS files have the data, and the priority. If they have the same CSS definition, the higher priority will be applied. You can also disable some of them to make sure the disabled style will no longer be applied into the site.

- For example: By default, a WCM demo package has two sites: ACME and Classic. The Classic site has a CSS folder which contains a CSS file called **DefaultStylesheet**. Most of the stylesheets of this site are defined within this stylesheet. Moreover, the ACME site has two CSS files called **BlueStylesheet** and

GreenStylesheet. The blue one is enabled and the green one is disabled by default. All you need to test is to disable the blue one (by editing it and setting Available equals false) and enable the green one. Now, back to the homepage and you will see the magic.

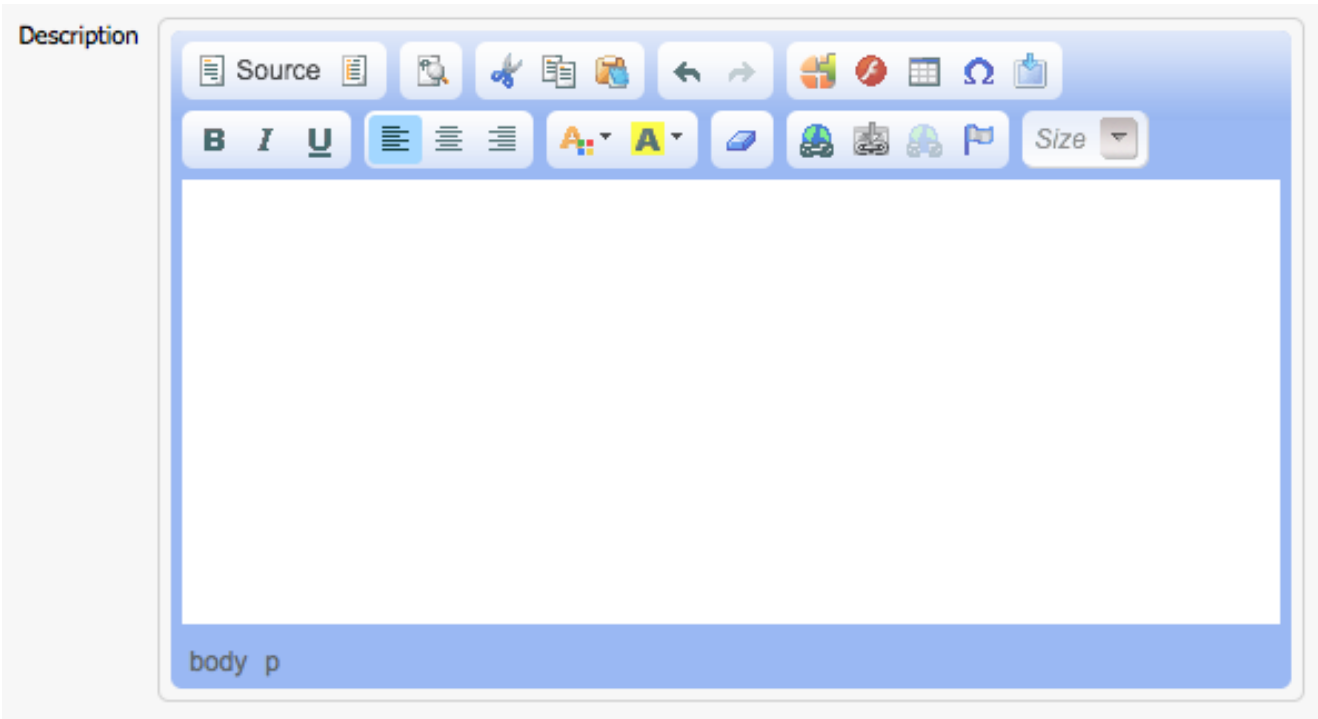
Note

Remember the cache and refresh the browser first if you do not see any changes. Normally, this is the main reason why the new style is not applied.

4.2.2. CKEditor

Basically, if you want to add a rich text area to your dialogs, you can use the addRichtextField method. However, in the case you want to add the rich text editor manually, you first need to use the addTextAreaField method and some additional Javascripts as shown below:

```
<%
        String[] fieldDescription = ["jcrPath=/node/exo:description"] ;
        uicomponent.addTextAreaField("description", fieldDescription)
%>
<script>
  var instances = CKEDITOR.instances['description'];
  if (instances) instances.destroy(true);
  CKEDITOR.replace('description', {
    toolbar : 'CompleteWCM',
    uiColor : '#9AB8F3'
  });
</script>
```



4.3. Extensions

4.3.1. REST Services

4.3.1.1. REST Overview

REST-style architectures consist of clients and servers. Clients initiate requests to servers; servers process requests and return appropriate responses. Requests and responses are built around the transfer of "representations" of "resources". A resource can be essentially any coherent and meaningful concept that may be addressed. A representation of a resource is typically a document that captures the current or intended state of a resource.

At any particular time, a client can either be in transition between application states or "at rest". A client in a REST state is able to interact with its users, but creates no load and consumes no per-client storage on the set of servers or on the network.

The client begins sending requests when it is ready to make the transition to a new state. While one or more requests are outstanding, the client is considered to be in transition. The representation of each application state contains links that may be used the next time, the client chooses to initiate a new state transition.

REST is initially described in the context of HTTP, but is not limited to that protocol. RESTful architectures can be based on other Application Layer protocols if they already provide a rich and uniform vocabulary for applications based on the transfer of meaningful representational state. RESTful applications maximize the use of the pre-existing, well-defined interface and other built-in capabilities provided by the chosen network protocol, and minimize the addition of new application-specific features on its top.

4.3.1.2. Restful Web Service

4.3.1.2.1. HTTP Methods

Here is the convention you should follow:

Method	Definition
GET	Gest a Resource. Its state should not be modified.
POST	Creates a Resource (or anything that does not fit elsewhere).
PUT	Updates a Resource.
DELETE	Deletes a Resource.

4.3.1.2.2. Formats

The followings is formats which need to be supported for all your APIs:

- Json: This format makes developers easy to parse in a lot of languages, such as JavaScript, Python or Ruby.
- XML: Most of Java developers like using this format.
- Atom: This is a standard format which can be used by many applications.

4.3.1.2.3. Data Format

The default format is JSON.

The response format can be specified by a parameter in the request: "*format*" Specify the format requested.

4.3.1.2.4. REST configuration

First, we have to register the REST service class to the configuration file in the package named **conf.portal** .

```
<configuration xmlns="http://www.exoplaform.org/xml/ns/kernel_1_1.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSche
  <component>
    <type>org.exoplatform.services.ecm.publication.REST.presentation.document.publication.PublicationGetDocument
  </component>
</configuration>
```

4.3.1.2.5. Create a REST service

You can start creating GetEditedDocumentRESTService that implements from the ResourceContainer interface as follows:

```
@Path("/presentation/document/edit/")
public class GetEditedDocumentRESTService implements ResourceContainer {
        @Path("/{repository}/")
                @GET
                public Response getLastEditedDoc(@PathParam("repository") String repository,
                   @QueryParam("showItems") String showItems) throws Exception {
                   ........
                }
}
```

Parameters	Definition
@Path("/presentation/document/edit/")	Specifies the URI path which a resource or class method will serve requests for.
@PathParam("repository")	Binds the value **repository** of a URI parameter or a path segment containing the template parameter to a resource method parameter, resource class field, or resource class bean property.
@QueryParam("showItems")	Binds the value **showItems** of a HTTP query parameter to a resource method parameter, resource class field, or resource class bean property.

4.3.2. UI Extensions

This part describes how to create a sample UI extension.

4.3.2.1. Overview

In WCM 2.2.0, it is possible to extend the **Content Explorer** and the **ECM Administration** with the **UI Extension Framework**. Indeed, you can add your own action buttons to the **Content Explorer** and/or add your own managers to the **ECM Administration**.

4.3.2.2. How to add your own tab in ECM Administration

4.3.2.2.1. Add your own UIAction

To add your own UIAction, do as follows:

1. Create a pom.xml file with the following content:

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:sch
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.exoplatform.ecms</groupId>
    <artifactId>exo-ecms-examples-uiextension-framework</artifactId>
    <version>2.2.0-SNAPSHOT</version>
  </parent>
  <artifactId>exo-ecms-examples-uiextension-framework-manage-wcm-cache</artifactId>
  <name>eXo WCM Cache Examples </name>
  <description>eXo WCM Cache Examples </description>
  <dependencies>
    <dependency>
      <groupId>org.exoplatform.kernel</groupId>
      <artifactId>exo.kernel.container</artifactId>
      <version>2.3.0-CR4-CP01</version>
      <scope>compile</scope>
    </dependency>
    <dependency>
      <groupId>org.exoplatform.commons</groupId>
      <artifactId>exo.platform.commons.webui.ext</artifactId>
      <version>1.1.0-GA</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.exoplatform.ecms</groupId>
      <artifactId>exo-ecms-core-webui</artifactId>
      <version>2.2.0-GA</version>
      <scope>provided</scope>
    </dependency>
    <dependency>
      <groupId>org.exoplatform.ecms</groupId>
      <artifactId>exo-ecms-core-webui-administration</artifactId>
      <version>2.2.0-GA</version>
      <scope>provided</scope>
    </dependency>
  </dependencies>
  <build>
    <resources>
      <resource>
        <directory>src/main/java</directory>
        <includes>
          <include>**/*.xml</include>
        </includes>
      </resource>
      <resource>
        <directory>src/main/resources</directory>
        <includes>
          <include>**/*.properties</include>
          <include>**/*.xml</include>
          <include>**/*.jar</include>
          <include>**/*.pom</include>
          <include>**/*.conf</include>
          <include>**/*.gtmpl</include>
          <include>**/*.gif</include>
          <include>**/*.jpg</include>
          <include>**/*.png</include>
        </includes>
      </resource>
    </resources>
  </build>
</project>
```

2. Create the *src/main/java* directory and start launching *mvn eclipse:eclipse_. Then, you can launch your eclipse and import this new project.*

3. Create a new class called **org.exoplatform.wcm.component.cache.UIWCMCacheComponent** that extends **org.exoplatform.ecm.webui.component.admin.manager.UIAbstractManagerComponent**.

4.3.2.2.2. Add your own ActionListener

With the Webui framework, you will be notified if any given action is triggered. You just need to call your own action listener ($ACTIONNAME) **ActionListener**. For example, to create your own action listener for **CacheView**, do as follows:

1. Call **CacheViewActionListener**.

2. Add a static inner class called **CacheViewActionListener** that extends **org.exoplatform.ecm.webui.component.admin.listener.UIECMAdminControlPanelActionListener** .

You will see the expected code below:

```
public class CacheViewComponent extends UIAbstractManagerComponent {
  public static class CacheViewActionListener extends UIECMAdminControlPanelActionListener<UIWCMCacheComponent>
    public void processEvent(Event<UIWCMCacheComponent> event) throws Exception {
    UIECMAdminPortlet portlet = event.getSource().getAncestorOfType(UIECMAdminPortlet.class);
       UIECMAdminWorkingArea uiWorkingArea = portlet.getChild(UIECMAdminWorkingArea.class);
       uiWorkingArea.setChild(UIWCMCachePanel.class) ;
       event.getRequestContext().addUIComponentToUpdateByAjax(uiWorkingArea);
    }
  }
}
```

3. Create the *src/main/java* directory and launch *mvn eclipse:eclipse_. You can then, launch your eclipse and import this new project.*

4. Create a new configuration file *conf/portal/configuration.xml* to register your action (see 6.2.2.3) with the service *org.exoplatform.webui.ext.UIExtensionManager*.

4.3.2.2.3. Register your UI Action

To register your UI Action, do as follows:

1. Create the code:

```
<configuration xmlns="http://www.exoplaform.org/xml/ns/kernel_1_1.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSche

  <external-component-plugins>
    <target-component>org.exoplatform.webui.ext.UIExtensionManager</target-component>
    <component-plugin>
      <name>Add.Actions</name>
      <set-method>registerUIExtensionPlugin</set-method>
      <type>org.exoplatform.webui.ext.UIExtensionPlugin</type>
      <init-params>
        <object-param>
          <name>CacheView</name>
          <object type="org.exoplatform.webui.ext.UIExtension">
            <field name="type">
              <string>org.exoplatform.ecm.dms.UIECMAdminControlPanel</string>
            </field>
            <field name="name">
              <string>CacheView</string>
            </field>
            <field name="category">
              <string>GlobalAdministration</string>
            </field>
            <field name="component">
              <string>org.exoplatform.wcm.component.cache.UIWCMCacheComponent</string>
            </field>
```

```
          </object>
        </object-param>
        <object-param>
          <name>UIWCMCacheManager</name>
          <object type="org.exoplatform.webui.ext.UIExtension">
            <field name="type">
              <string>org.exoplatform.ecm.dms.UIECMAdminControlPanel</string>
            </field>
            <field name="name">
              <string>UIWCMCacheManager</string>
            </field>
            <field name="category">
              <string>GlobalAdministration</string>
            </field>
            <field name="component">
              <string>org.exoplatform.wcm.manager.cache.UIWCMCacheManagerComponent</string>
            </field>
            <field name="extendedFilters">
               <collection type="java.util.ArrayList">
                  <value>
                    <object type="org.exoplatform.webui.ext.filter.impl.UserACLFilter">
                       <field name="permissions">
                          <collection item-type="java.lang.String" type="java.util.ArrayList">
                             <value>
                                <string>*:/platform/administrators</string>
                              </value>
                          </collection>
                       </field>
                    </object>
                  </value>
               </collection>
            </field>
          </object>
        </object-param>
      </init-params>
    </component-plugin>
  </external-component-plugins>
</configuration>
```

Note

With this configuration, only the users with the **:/platform/administrators* membership have the right to access the CacheView item.

2. Launch **mvn clean install**.

3. Copy the resource bundle

Note

All resources can be located in the *src/main/resource* package because the resources (*.xml, images, conf file) and the code are separated. This is very useful in a hierarchical structure.

Create *ExamplePortlet*en.xml *with the following content and add it to* src/main/resource *package:*

```
<bundle>
  <!-- ############################################################################## # org.exoplatform.wcm.co
    # ############################################################################## -->

  <UIWCMCacheForm>
    <action>
      <Cancel>Cancel</Cancel>
      <Save>Save</Save>
      <Clear>Clear the cache</Clear>
    </action>
    <label>
      <maxsize>Max size :</maxsize>
```

```
      <livetime>Live time in sec :</livetime>
      <isCacheEnable>Cache enabled(should always be on production enviroment)</isCacheEnable>
      <hit>Hit count :</hit>
      <currentSize>Current size</currentSize>
      <miss>Miss count :</miss>
    </label>
  </UIWCMCacheForm>

  <!-- ################################################################################### # org.exoplatform.wcm.ma
    # ################################################################################### -->

  <UIWCMCacheManagerForm>
    <action>
      <Cancel>Cancel</Cancel>
      <Save>Save</Save>
      <Clear>Clear the cache</Clear>
    </action>
    <label>
      <cacheModify>Cache to modify :</cacheModify>
      <maxsize>Max size :</maxsize>
      <livetime>Live time in sec :</livetime>
      <isCacheEnable>Cache enabled(should always be on production enviroment)</isCacheEnable>
      <hit>Hit count :</hit>
      <currentSize>Current size</currentSize>
      <miss>Miss count :</miss>
    </label>
  </UIWCMCacheManagerForm>

  <UIECMAdminControlPanel>
    <tab>
      <label>
        <GlobalAdministration>Global Administration</GlobalAdministration>
      </label>
    </tab>
    <label>
      <UIWCMCache>WCM Cache</UIWCMCache>
      <UIWCMCachePanel>WCM Cache Administration</UIWCMCachePanel>
      <UIWCMCacheManager>Managing Caches</UIWCMCacheManager>
      <UIWCMCacheManagerPanel>WCM Cache Management</UIWCMCacheManagerPanel>
    </label>
  </UIECMAdminControlPanel>
</bundle>
```

You must add the following content to *configuration.xml- to register the resource bundle.*

Note

By being added this configuration, the resource bundle has been completely separated from the original system, this is so clearly useful, you get an independent plugin.

```
<external-component-plugins>
  <!-- The full qualified name of the ResourceBundleService -->
  <target-component>org.exoplatform.services.resources.ResourceBundleService</target-component>
  <component-plugin>
    <!-- The name of the plugin -->
    <name>ResourceBundle Plugin</name>
    <!-- The name of the method to call on the ResourceBundleService in order to register the ResourceBundles --
    <set-method>addResourceBundle</set-method>
    <!-- The full qualified name of the BaseResourceBundlePlugin -->
    <type>org.exoplatform.services.resources.impl.BaseResourceBundlePlugin</type>
    <init-params>
      <values-param>
        <name>init.resources</name>
        <description>Store the following resources into the db for the first launch </description>
        <value>locale.portlet.cache.ExamplePortlet</value>
      </values-param>
      <values-param>
        <name>portal.resource.names</name>
        <description>The properties files of the portal , those file will be merged
          into one ResoruceBundle properties </description>
        <value>locale.portlet.cache.ExamplePortlet</value>
      </values-param>
```

```
    </init-params>
  </component-plugin>
</external-component-plugins>
```

4.3.2.2.4. Run your own UI extension sample

- Run tomcat
- Sign in as root
- Go to the ECM Administration.

You will see that a new extension has been already added to ECM Administrator:

4.3.3. Authoring Extension

Publication addons for WCM 2.2.0

4.3.3.1. Extended Publication Plugin

4.3.3.1.1. States

This extended publication has new states and new profiles that are enable in WCM 2.2.0.

- Profiles
  - Author: This profile can edit a content and mark this content as redacted.
  - Approver: This profile approves a pending content (marked by the Author).
  - Publisher: This profile publishes contents or marks them as "Ready for publication" in multi-server mode.
  - Archiver: An administrative profile which moves contents to an archive storage.
- States
  - enrolled: It is a pure technical state, generally used for content creation.
  - draft (Author): Content is in editing phase.
  - pending (Author): The author validates the content.
  - approved (Approver): A content is approved by the manager.
  - inreview (Manager): This state can be used when a second approval state is needed (for i18 translation for example).
  - staged (Publisher): A content is ready for publication (multi-server mode).
  - published (Publisher or Automatic): A content is published and visible in the **Live** mode.
  - unpublished (Publisher or Automatic): A content is not visible in the **Live** mode.
  - obsolete: A content can still be published but it is not in an editing lifecycle anymore.

- archived (Automatic): A content is archived and ready to be moved in the archive workspace if enabled.

4.3.3.1.2. Start/End publication dates

In most cases, you do not want to publish a content directly, but at a defined date and you can also want to unpublish automatically the content after that. New properties are added to the new publication plugin, that allows you to manage this:

- publication:startPublishedDate
- publication:endPublishedDate

The WCM 2.2.0 rendering engine does not know anything about publication dates, so another service needs to manage that. When the publisher sets start/end publication dates, he can "stage" the content. The content will go automatically to "published" state when the start date arrives and to "unpublished" state after end date. A cron job checks every hour (or less) all contents which need to be published (start date in the past + "staged" state) or unpublished (end date in the past + "published" state).

Because of that, publication dates are not mandatory and a content can go to:

- Staged : in multi-server mode, the publisher can only put the content to the "staged" state and wait for auto-publication.
- Published : in single-server mode, the publisher can directly publish a content (with or without publication dates).

4.3.3.1.3. New Publication Mixin

```
<nodeType hasOrderableChildNodes="false" isMixin="true" name="publication:authoringPublication" primaryItemName=
  <supertypes>
    <supertype>publication:stateAndVersionBasedPublication</supertype>
  </supertypes>
  <propertyDefinitions>
    <propertyDefinition autoCreated="false" mandatory="true" multiple="false" name="publication:startPublishedDa
      <valueConstraints/>
    </propertyDefinition>
    <propertyDefinition autoCreated="false" mandatory="true" multiple="false" name="publication:endPublishedDate
      <valueConstraints/>
    </propertyDefinition>
  </propertyDefinitions>
</nodeType>
```

4.3.3.1.3.1. Publication plugin UI

Note that some labels like "Pr#t # publier" could be not well displayed in the publication UI. You can extend the width of the current UI State button by adding:

```
.UIPublicationPanel .StatusTable .ActiveStatus {
  width: 75px !important;
}
```

Another quick note about publication date inputs. UIPublicationPanel should not initialize the dates to any default value. The publishing and unpublish cron jobs will do this:

- A staged document with null publication start date is published instantly.
- A document with null publication end date is published forever.

See export section for more info about cron jobs.

4.3.3.2. Publication Manager

The Publication Manager manages lifecycles and contexts in the WCM platform. It allows to manages different lifecycles based on different publication plugin in the platform.

```
public interface PublicationManager {

        public List<Lifecycle> getLifecycles();

        public List<Context> getContexts();

        public Context getContext(String name);

        public Lifecycle getLifecycle(String name);

        public List<Lifecycle> getLifecyclesFromUser(String remoteUser, String state);
}
```

- getLifecycles: returns a list of lifecycles (see below), with lifecycle name, publication plugin involved and possible states.
- getContexts: returns a list of context, with name, related Lifecycle and other properties (see below).
- getContext: returns a context by its name.
- getLifecycle: returns a lifecycle by its name.
- getLifecycleFromUser: returns a list of lifecycles in which the user has rights (based on membership property).

4.3.3.2.1. Lifecycle

A lifecycle is defined by a simple vertical workflow with steps (states) and profiles (membership). Each lifecycle is related to a Publication plugin (to be compliant with JBPM or Bonita business processes). Example: *Two lifecycles with/without states*

```
<external-component-plugins>
  <target-component>org.exoplatform.services.wcm.publication.PublicationManager</target-component>
  <component-plugin>
    <name>AddLifecycle</name>
    <set-method>addLifecycle</set-method>
    <type>org.exoplatform.services.wcm.publication.lifecycles.StatesLifecyclePlugin</type>
    <init-params>
      <object-param>
        <name>lifecyles</name>
        <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig">
          <field name="lifecycles">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$Lifecycl
                  <field name="name">
                    <string>lifecycle1</string>
                  </field>
                  <field name="publicationPlugin">
                    <string>States and versions based publication</string>
                  </field>
```

```
        </object>
      </value>
      <value>
        <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$Lifecycl
          <field name="name">
            <string>lifecycle2</string>
          </field>
          <field name="publicationPlugin">
            <string>Authoring publication</string>
          </field>
          <field name="states">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                  <field name="state">
                    <string>draft</string>
                  </field>
                  <field name="memberships">
                    <collection type="java.util.ArrayList">
                      <value>
                        <string>author:/CA/communicationDG</string>
                      </value>
                      <value>
                        <string>author:/CA/alerteSanitaire</string>
                      </value>
                      <value>
                        <string>author:/CA/alerteInformatique</string>
                      </value>
                      <value>
                        <string>author:/CA/informations</string>
                      </value>
                    </collection>
                  </field>
                </object>
              </value>
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                  <field name="state">
                    <string>pending</string>
                  </field>
                  <field name="membership">
                    <string>author:/platform/web-contributors</string>
                  </field>
                </object>
              </value>
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                  <field name="state">
                    <string>approved</string>
                  </field>
                  <field name="membership">
                    <string>manager:/platform/web-contributors</string>
                  </field>
                </object>
              </value>
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                  <field name="state">
                    <string>staged</string>
                  </field>
                  <field name="membership">
                    <string>publisher:/platform/web-contributors</string>
                  </field>
                </object>
              </value>
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                  <field name="state">
                    <string>published</string>
                  </field>
                  <field name="membership">
                    <string>automatic</string>
                  </field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </value>
```

```
              <value>
                <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$Lifecycl
                  <field name="name">
                    <string>lifecycle3</string>
                  </field>
                  <field name="publicationPlugin">
                    <string>Authoring publication</string>
                  </field>
                  <field name="states">
                    <collection type="java.util.ArrayList">
                      <value>
                        <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                          <field name="state">
                            <string>draft</string>
                          </field>
                          <field name="membership">
                            <string>author:/platform/web-contributors</string>
                          </field>
                        </object>
                      </value>
                      <value>
                        <object type="org.exoplatform.services.wcm.publication.lifecycles.impl.LifecyclesConfig$
                          <field name="state">
                            <string>published</string>
                          </field>
                          <field name="memberships">
                            <collection type="java.util.ArrayList">
                              <value>
                                <string>publisher:/CA/communicationDG</string>
                              </value>
                              <value>
                                <string>publisher:/CA/alerteSanitaire</string>
                              </value>
                              <value>
                                <string>publisher:/CA/alerteInformatique</string>
                              </value>
                              <value>
                                <string>publisher:/CA/informations</string>
                              </value>
                            </collection>
                          </field>
                        </object>
                      </value>
                    </collection>
                  </field>
                </object>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

In the last example, there are three lifecycles:

- Lifecycle 1: based on *States and versions based publication*.
  - This allows to be backward compliant with older WCM releases. If all your site contents are using an existing plugin, you can create a lifecycle for it and it will work.
  - For new instances, you should use the new plugin with dynamic states capabilities.
- Lifecycle 2: based on new *Authoring publication*.
  - Visibility: Define only the "visible" steps. In this example, there is no step for 'enrolled'. Even if this step exists, it will not be displayed in the UI.
  - Automatic: Set a step as "automatic". In this mode, the step will be visible in the UI but it will be managed

by the system (a cron job for example).

- Lifecycle 3: Simulates the *States and versions based publication* plugin.
  - Note that this simple lifecycle will work in a single server configuration.

4.3.3.2.1.1. Listen to a lifecycle

When a state is changed, you can broadcast an event to add features. The event could look like this:

```
listenerService.broadcast(AuthoringPlugin.POST_UPDATE_STATE_EVENT, null, node);
```

Listener declaration could look like this:

```
<external-component-plugins>
  <target-component>org.exoplatform.services.listener.ListenerService</target-component>
  <component-plugin>
    <name>PublicationService.event.postUpdateState</name>
    <set-method>addListener</set-method>
    <type>org.exoplatform.services.wcm.publication.listener.post.PostUpdateStateEventListener</type>
    <description>this listener will be called every time a content changes its current state</description>
  </component-plugin>
</external-component-plugins>
```

4.3.3.2.2. Context

A context is defined by simple rules. In WCM 2.2.0, you can choose to enroll the content in a specific lifecycle (e.g. publication plugin) based on context parameters. There are three parameters used to define contexts:

- Remote User: The current user who create/edit the content.
- Current sitename: The Site from where the content is created (not the storage but the navigation).
- Node: The node which you want to enroll.

From these parameters, you can easily connect and define contexts based on:

- Membership: Does the current user have this membership?
- Site: On this particular site, you want to enroll contents in a specific lifecycle.
- Path: You can enroll contents in the lifecycles based on their path (from the Node).
- Type of content: You can enroll contents in the lifecycles based on their nodetype (from the Node).

Because each site has a content storage (categories + physical storage), you can choose the right lifecycle for the right storage/site. Conflicts on contexts can occurr, you should avoid it by setting a priority (The less is the best).

Example : *Different Contexts*

```
<external-component-plugins>
  <target-component>org.exoplatform.services.wcm.publication.PublicationManager</target-component>
  <component-plugin>
    <name>AddContext</name>
```

```
    <set-method>addContext</set-method>
    <type>org.exoplatform.services.wcm.publication.context.ContextPlugin</type>
    <init-params>
      <object-param>
        <name>contexts</name>
        <object type="org.exoplatform.services.wcm.publication.context.impl.ContextConfig">
          <field name="contexts">
            <collection type="java.util.ArrayList">
              <value>
                <object type="org.exoplatform.services.wcm.publication.context.impl.ContextConfig$Context">
                  <field name="name">
                    <string>contextdefault</string>
                  </field>
                  <field name="priority">
                    <string>200</string>
                  </field>
                  <field name="lifecycle">
                    <string>lifecycle1</string>
                  </field>
                </object>
                <object type="org.exoplatform.services.wcm.publication.context.impl.ContextConfig$Context">
                  <field name="name">
                    <string>context1</string>
                  </field>
                  <field name="priority">
                    <string>100</string>
                  </field>
                  <field name="lifecycle">
                    <string>lifecycle1</string>
                  </field>
                  <field name="membership">
                    <string>*:/platform/web-contributors</string>
                  </field>
                  <field name="site">
                    <string>acme</string>
                  </field>
                  <field name="path">
                    <string>repository:collaboration:/sites content/live/acme/categories</string>
                  </field>
                </object>
                <object type="org.exoplatform.services.wcm.publication.context.impl.ContextConfig$Context">
                  <field name="name">
                    <string>context2</string>
                  </field>
                  <field name="priority">
                    <string>100</string>
                  </field>
                  <field name="lifecycle">
                    <string>lifecycle1</string>
                  </field>
                  <field name="site">
                    <string>classic</string>
                  </field>
                </object>
                <object type="org.exoplatform.services.wcm.publication.context.impl.ContextConfig$Context">
                  <field name="name">
                    <string>context3</string>
                  </field>
                  <field name="priority">
                    <string>80</string>
                  </field>
                  <field name="lifecycle">
                    <string>lifecycle3</string>
                  </field>
                  <field name="membership">
                    <string>manager:/company/finances</string>
                  </field>
                  <field name="path">
                    <string>repository:collaboration:/documents/company/finances</string>
                  </field>
                </object>
                <object type="org.exoplatform.services.wcm.publication.context.impl.ContextConfig$Context">
                  <field name="name">
                    <string>context4</string>
                  </field>
                  <field name="priority">
                    <string>50</string>
                  </field>
                  <field name="lifecycle">
```

```
                  <string>lifecycle4</string>
                </field>
                <field name="memberships">
                  <collection type="java.util.ArrayList">
                    <value>
                      <string>manager:/CA/communicationDG</string>
                    </value>
                    <value>
                      <string>manager:/CA/alerteSanitaire</string>
                    </value>
                    <value>
                      <string>manager:/CA/alerteInformatique</string>
                    </value>
                    <value>
                      <string>manager:/CA/informations</string>
                    </value>
                  </collection>
                </field>
                <field name="path">
                  <string>repository:collaboration:/documents/company/finances</string>
                </field>
                <field name="nodetype">
                  <string>exo:article</string>
                </field>
              </object>
            </value>
          </collection>
        </field>
      </object>
    </object-param>
  </init-params>
 </component-plugin>
</external-component-plugins>
```

The logic is very simple. When creating a content, it should be attached a lifecycle with the lifecyle priority:

- *context4* is the most important (priority=50): you will enroll the content in the lifecycle 'lifecycle4' if:
  - The content creator has the 'manager:/company/finances' membership.
  - The content is stored in 'repository:collaboration:/documents/company/finances' or any subfolders.
  - The content is a 'exo:article'.
- If not, you will continue with *context3*.
- etc.

The logic is very simple. When we create a content, we will go lifecycle by lifecycle starting with the better priority :

- *context4* is the most important (priority=50) : we will enroll the content in the lifecycle 'lifecycle4' if :
  - the content creator has the 'manager:/company/finances' membership
  - the content is stored in 'repository:collaboration:/documents/company/finances' or any subfolders
  - the content is a 'exo:article'
- if not we will continue with *context3*
- etc

Note

Ihe contexts will be used only when the content is created and when you want to enroll it in a lifecycle for the first time. Once you have the corresponding lifecycle, you will set the lifecycle inside the content (see New authoring Mixin) and the context service will not be called again for this content.

4.3.3.2.3. New Authoring Mixin

```
<nodeType hasOrderableChildNodes="false" isMixin="true" name="publication:authoring" primaryItemName="">
  <propertyDefinitions>
    <propertyDefinition autoCreated="false" mandatory="false" multiple="false" name="publication:lastUser" onPar
      <valueConstraints/>
    </propertyDefinition>
    <propertyDefinition autoCreated="false" mandatory="false" multiple="false" name="publication:lifecycle" onPa
      <valueConstraints/>
    </propertyDefinition>
  </propertyDefinitions>
</nodeType>
```

When adding the content in a lifecycle, set the *publication:lifecycle_ property with the corresponding lifecycle.*

Note

A content can be in one lifecycle only.

Each time you change from one state to another, you set the user who changed the state in *publication:lastUser_*.

4.3.3.2.3.1. Querying based on publication status

By adding this mixin to contents, you can access contents by simple queries based on the current user profile. For example:

- All my draft contents:
  - query: select * from nt:base where publication:currentState='draft' and publication:lastUser='benjamin';
- All the contents I have to approve
  - call: PublicationManager.getLifecycles('benjamin', 'approved') => returns lifecycles where I can go to the 'approved' state.
  - query: select * from nt:base where publication:currentState='pending' and (publication:lifecycle='lifecycle1' or publication:lifecycle='lifecycle3');
- All the content that will be published tomorrow
  - query: select * from nt:base where publication:currentState='staged' and publication:startPublishedDate>'xxxx';

4.4. Public REST APIs

4.4.1. ThumbnailRESTService

Service name	Service URL	Location	Description
ThumbnailRESTService	{portalname}/{restcontextname}/thumbnailImage/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-services	Returns a responding data as a thumbsnail image.

Note

- {portalname}: The name of the portal.
- {restcontextname}: The context name of rest webapplication which is deployed to the "{portalname}" portal.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getThumbnailImage	{portalname}/{restcontextname}/thumbnailImage/medium/{repoName}/{workspaceName}/{nodePath}	repoName workspaceName nodePath	String String String	Returns a responding data with the medium size (64x64).
getLargeImage	{portalname}/{restcontextname}/thumbnailImage/large/{repoName}/{workspaceName}/{nodePath}	repoName workspaceName nodePath	String String String	Returns a responding data with the large size (300x300).
getSmallImage	{portalname}/{restcontextname}/thumbnailImage/small/{repoName}/{workspaceName}/{nodePath}	repoName workspaceName nodePath	String String String	Returns a responding data with the small size (32x32).
getCustomImage	{portalname}/{restcontextname}/thumbnailImage/custom/{size}/{repoName}/{workspaceName}/{	size repoName	String String	Returns a responding data with the custom size.

Name	Service URL endpoint	Parameters	Expected Values	Description
		workspaceName nodePath	String String	
getOriginImage	{portalname}/{restcontextname}/thumbnailImage/origin/{repoName}/{workspaceName}/{nodePath}	repoName workspaceName nodePath	String String String	Returns the image with the original size.

4.4.2. RssConnector

Service name	Service URL	Location	Description
RssConnector	{portalname}/{restcontextname}/feed/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-connector	Generates an RSS feed.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
generate	{portalname}/{restcontextname}/feed/rss/	repository workspace server siteName title desc folderPath	String String String String String String String	Generates an RSS feed.

Name	Service URL endpoint	Parameters	Expected Values	Description
		orderBy	String	
		orderType	String	
		lang	String	
		detailPage	String	
		detailParam	String	
		recursive	String	

4.4.3. FCKCoreRESTConnector

Service name	Service URL	Location	Description
FCKCoreRESTConnector	{portalname}/{restcontextname}//fckconnector/jcr/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-connector	Gets a list of files and folders, creates a folder and uploads files.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getFoldersAndFiles	{portalname}/{restcontextname}/fckconnector/jcr/getFoldersAndFiles/{repositoryName}/{workspa	repositoryName	String	Gets the folders and the files in the current folder.
		workspaceName	String	
		currentFolder	String	
		command	String	
		type	String	
createFolder	{portalname}/{restcontextname}/fckconnector/jcr/createFolder/{repositoryName}/{workspaceName	repositoryName	String	Creates a folder under the current folder.

Name	Service URL endpoint	Parameters	Expected Values	Description
		workspaceName	String	
		currentFolder	String	
		newFolderName	String	
		language	String	
uploadFile	{portalname}/{restcontextname}/fckconnector/jcr/uploadFile/	servletRequest	HttpServletRequest	Uploads a file with the HttpServletRequest.
processUpload	{portalname}/{restcontextname}/fckconnector/jcr/uploadFile/control/{repositoryName}/{workspace	repositoryName	String	Controls the process of uploading a file, such as aborting, deleting or progressing the file.
		workspaceName	String	
		currentFolder	String	
		action	String	
		language	String	
		fileName	String	
		uploadId	String	

4.4.4. ResourceBundleConnector

Service name	Service URL	Location	Description
ResourceBundleConnector	{portalname}/{restcontextname}/bundle/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-connector	Gets the bundle basing on the key and the locale.

exo-ecms-core-connector

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getBundle	{portalname}/{restcontextname}/bundle/getBundle/{key}/{locale}/	key locale	String String	Gets the bundle basing on the key and the locale.

4.4.5. VoteConnector

Service name	Service URL	Location	Description
VoteConnector	{portalname}/{restcontextname}/contents/vote/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-connector	Returns and sets a vote value of a given node in the sent parameter.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
postVote	{portalname}/{restcontextname}/contents/vote/postVote/{repositoryName}/{workspaceName}/{jcrPath}/{vote}/{lang}/	repositoryName workspaceName jcrPath vote lang	String String String String String String	Sets a vote value for a given content.
getVote	{portalname}/{restcontextname}/contents/vote/getVote/{repositoryName}/{workspaceName}/{jcrPath}/	repositoryName workspaceName jcrPath	String String String	Returns a vote value for a given content.

4.4.6. DriverConnector

Service name	Service URL	Location	Description
DriverConnector	{portalname}/{restcontextname}/wcmDriver/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-connector	Returns a drive list, a folder list and a document list in a specified location for a given user. Also, it processes the file uploading action.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getDrivers	{portalname}/{restcontextname}/wcmDriver/getDrivers/		String	Returns a list of drives for the current user.
getFoldersAndFiles	{portalname}/{restcontextname}/wcmDriver/getFoldersAndFiles/	driverName currentFolder currentPortal repositoryName workspaceName filterBy	String String String String String String	Returns all folders and files in a given location.
uploadFile	{portalname}/{restcontextname}/wcmDriver/uploadFile/upload/	uploadId	String	Uploads a file.
processUpload	{portalname}/{restcontextname}/wcmDriver/uploadFile/control/	repositoryName workspaceName driverName currentFolder currentPortal userId	String String String String String String	Controls the process of uploading a file, such as aborting, deleting or processing the file.

Name	Service URL endpoint	Parameters	Expected Values	Description
		jcrPath action language fileName uploadId	String String String String String	

4.4.7. GadgetConnector

Service name	Service URL	Location	Description
GadgetConnector	{portalname}/{restcontextname}/wcmGadget/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-connector	Instantiates a new gadget connector.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getFoldersAndFiles	{portalname}/{restcontextname}/wcmGadget/getFoldersAndFiles/{currentFolder}/{lang}/{host}/	currentFolder lang host	String String String	Gets the folders and files.

4.4.8. PortalLinkConnector

Service name	Service URL	Location	Description
PortalLinkConnector	{portalname}/{restcontextname}/portalLinks/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**:	Returns a page URI for a given location.

Service name	Service URL	Location	Description
		exo-ecms-core-connector	

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getPageURI	{portalname}/{restcontextname}/portalLinks/getFoldersAndFiles/	currentFolder	String String String	Gets the page URI.

4.4.9. GetEditedDocumentRESTService

Service name	Service URL	Location	Description
GetEditedDocumentRESTService	{portalname}/{restcontextname}/presentation/document/edit/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-publication	Returns the latest edited documents.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getLastEditedDoc	{portalname}/{restcontextname}/presentation/document/edit/{repository}/	{repository}	String	Returns the latest edited documents.

4.4.10. PublicationGetDocumentRESTService

Service name	Service URL	Location	Description
PublicationGetDocumentRESTService	{portalname}/{restcontextname}/publication/presentation/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-publication	Returns a list of published documents.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getPublishDocument	{portalname}/{restcontextname}/publication/presentation/{repository}/{workspace}/{state}/	repository workspace state showItems	String String String String	Returns a list of published document by the default plugin.
getPublishedListDocument	{portalname}/{restcontextname}/publication/presentation/{repository}/{workspace}/{publicationPl	repository workspace publicationPluginName state showItems	String String String String String String	Returns a list of published documents by a specific plugin.

4.4.11. FavoriteRESTService

Service name	Service URL	Location	Description
FavoriteRESTService	{portalname}/{restcontextname}/favorite/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-services	Returns a list of favourite documents of a given user.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
getFavoriteByUser	{portalname}/{restcontextname}/favorite/all/{repoName}/{workspaceName}/{userName}	repoName	String	Returns a list of favourite documents of a given user.

Name	Service URL endpoint	Parameters	Expected Values	Description
		workspaceName userName showItems	String String String	

4.4.12. RESTImagesRendererService

Service name	Service URL	Location	Description
RESTImagesRendererService	{portalname}/{restcontextname}/images/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-core-services	Gets the image binary data of a given image node.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
serveImage	{portalname}/{restcontextname}/images/{repositoryName}/{workspaceName}/{nodeIdentifier}/	repositoryName workspaceName nodeIdentifier param	String String String String	Gets the binary data of a given image node.

4.4.13. LifecycleConnector

Service name	Service URL	Location	Description
LifecycleConnector	{portalname}/{restcontextname}/authoring/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-ext-authoring-services	Returns a list of contents in a given state range of the publication lifecycle.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
byState	{portalname}/{restcontextname}/authoring/bystate/	fromstate user lang workspace json	String String String String String	Returns a list of contents from the given state to the last state.
toState	{portalname}/{restcontextname}/authoring/toState/	fromstate tostate user lang workspace json	String String String String String String	Returns a list of contents from the begining state to the last state.
byDate	{portalname}/{restcontextname}/authoring/byDate/	fromstate date lang workspace json	String String String String String String	Returns a list of contents from the given begining state and published before the given date.

4.4.14. CopyContentFile

Service name	Service URL	Location	Description
CopyContentFile	{portalname}/{restcontextname}/copyfile/	**Maven groupId**: org.exoplatform.ecms **ArtifactId**: exo-ecms-ext-authoring-services	Copies a file.

- **APIs usage**:

Name	Service URL endpoint	Parameters	Expected Values	Description
copyFile	{portalname}/{restcontextname}/copyfile/	param: copy information	String	Copies a file.

4.4.15. PDFViewerRESTService

4.5. Public Java APIs

4.5.1. TaxonomyService

Taxonomy service is used to work with taxonomies. In this service, there are many functions which enable you to add, find, delete taxonomies from a node.

Method	Return	Prototype	Description
getTaxonomyTree	`Node`	`getTaxonomyTree(String repository, String taxonomyName, boolean system) throws RepositoryException;`	Returns the root node of the given taxonomy tree. @param repository: The name of repository. @param taxonomyName: The name of the taxonomy. @param system: Indicates whether the nodes must be retrieved using a session system or user session.

Method	Return	Prototype	Description
			@throws RepositoryException: if the taxonomy tree could not be found.
getTaxonomyTree	`Node`	`getTaxonomyTree(String repository, String taxonomyName) throws RepositoryException;`	Returns the root node of the given taxonomy tree with the user session. @param repository: The name of repository. @param taxonomyName: The name of the taxonomy. @throws RepositoryException: if the taxonomy tree could not be found.
getAllTaxonomyTrees	`List<Node>`	`getAllTaxonomyTrees(String repository, boolean system) throws RepositoryException;`	Returns the list of all the root nodes of the taxonomy tree available. @param repository: The name of repository. @param system: Indicates whether the nodes must be retrieved using a session system or user session. @throws RepositoryException: if the taxonomy trees could not be found .
getAllTaxonomyTrees	`List<Node>`	`getAllTaxonomyTrees(String repository) throws RepositoryException;`	Returns the list of all the root nodes of the taxonomy tree available with the user session. @param repository: The

Method	Return	Prototype	Description
			name of repository. @throws RepositoryException: if the taxonomies could not be found.
hasTaxonomyTree	`boolean`	`hasTaxonomyTree(String repository, String taxonomyName) throws RepositoryException;`	Checks if a taxonomy tree with the given name has already been defined. @param repository: The name of repository. @param taxonomyName: The name of the taxonomy. @throws RepositoryException: if the taxonomy name could not be checked.
addTaxonomyTree	`void`	`addTaxonomyTree(Node taxonomyTree) throws RepositoryException, TaxonomyAlreadyExistsException;`	Defines a node as a new taxonomy tree. @param taxonomyTree: The taxonomy tree to define. @throws TaxonomyAlreadyExistsException: if a taxonomy with the same name has already been defined. @throws RepositoryException: if the taxonomy tree could not be defined.
updateTaxonomyTree	`void`	`updateTaxonomyTree(String taxonomyName, Node taxonomyTree) throws RepositoryException;`	Re-defines a node as a taxonomy tree. @param taxonomyName:

Method	Return	Prototype	Description
			The name of the taxonomy to update. @param taxonomyTree: The taxonomy tree to define. @throws RepositoryException: if the taxonomy tree could not be updated.
removeTaxonomyTree	`void`	`removeTaxonomyTree(String taxonomyName) throws RepositoryException;`	Remove the taxonomy tree definition. @param taxonomyName: The name of the taxonomy to remove. @throws RepositoryException: if the taxonomy tree could not be removed.
addTaxonomyNode	`void`	`addTaxonomyNode(String repository, String workspace, String parentPath, String taxoNodeName, String creator) throws RepositoryException, TaxonomyNodeAlreadyExistsException;`	Adds a new taxonomy node at the given location. @param repository: The name of the repository. @param workspace: The name of the workspace @param parentPath: The place where the taxonomy node will be added. @param taxoNodeName: The name of taxonomy node.

Method	Return	Prototype	Description
			@param creator: The name of the user creating this node. @throws TaxonomyNodeAlreadyExistsExceptio if a taxonomy node with the same name has already been added. @throws RepositoryException: if the taxonomy node could not be added.
removeTaxonomyNode	`void`	`removeTaxonomyNode(String repository, String workspace, String absPath) throws RepositoryException;`	Removes the taxonomy node located at the given absolute path. @param repository: The name of the repository @param workspace: The name of the workspace. @param absPath: The absolute path of the taxonomy node to remove. @throws RepositoryException: if the taxonomy node could not be removed.
moveTaxonomyNode	`void`	`moveTaxonomyNode(String repository, String workspace, String srcPath, String destPath, String type) throws RepositoryException;`	Copies or cuts the taxonomy node from source path to destination path. The parameter type indicates if the node must be cut or copied. @param repository: The name of the repository.

Method	Return	Prototype	Description
			@param workspace: The name of the workspace. @param srcPath: The source path of this taxonomy. @param destPath: The destination path of the taxonomy. @param type: If **type** is equal to **cut**, the process will be cut. If **type** is equal to **copy**, the process will be copied. @throws RepositoryException: if the taxonomy node could not be moved.
hasCategories	`boolean`	`hasCategories(Node node, String taxonomyName) throws RepositoryException;`	Returns true if the given node has categories in the given taxonomy. @param node: The node to check. @param taxonomyName: The name of the taxonomy. @throws RepositoryException: if categories cannot be checked.
hasCategories	`boolean`	`hasCategories(Node node, String taxonomyName, boolean system) throws RepositoryException;`	Return true if the given node has categories in the given taxonomy. @param node: The node to check.

Method	Return	Prototype	Description
			@param taxonomyName: The name of the taxonomy. @param system: check system provider or not. @throws RepositoryException: if categories cannot be checked.
getCategories	`List<Node>`	`getCategories(Node node, String taxonomyName) throws RepositoryException;`	Returns all the paths of the categories (relative to the root node of the given taxonomy) which have been associated to the given node for the given taxonomy. @param node: The node for which we seek the categories. @param taxonomyName: The name of the taxonomy. @throws RepositoryException: if the categories cannot be retrieved.
getCategories	`List<Node>`	`getCategories(Node node, String taxonomyName, boolean system) throws RepositoryException;`	Returns all the paths of the categories(relative to the root node of the given taxonomy) which have been associated to the given node for the given taxonomy. @param node: The node for which we seek the categories. @param taxonomyName:

Method	Return	Prototype	Description
			The name of the taxonomy. @param system. @throws RepositoryException: if the categories cannot be retrieved.
getAllCategories	`List<Node>`	`getAllCategories(Node node) throws RepositoryException;`	Returns all the paths of the categories which have been associated to the given node. @param node: The node for which we seek the categories @throws RepositoryException.
getAllCategories	`List<Node>`	`getAllCategories(Node node, boolean system) throws RepositoryException;`	Returns all the paths of the categories which have been associated to the given node. @param node: The node for which we seek the categories @param system: check system provider or not . @throws RepositoryException.
removeCategory	`void`	`removeCategory(Node node, String taxonomyName, String categoryPath) throws RepositoryException;`	Removes a category to the given node. @param node: The node from which we remove the category.

Method	Return	Prototype	Description
			@param taxonomyName: The name of the taxonomy. @param categoryPath: The path of the category relative to the root node of the given taxonomy. @throws RepositoryException: if the category cannot be removed.
removeCategory	`void`	`removeCategory(Node node, String taxonomyName, String categoryPath, boolean system) throws RepositoryException;`	Removes a category to the given node. @param node: The node from which we remove the category. @param taxonomyName: The name of the taxonomy. @param categoryPath: The path of the category relative to the root node of the given taxonomy. @param system: check system provider or not. @throws RepositoryException: if the category cannot be removed.
addCategories	`void`	`addCategories(Node node, String taxonomyName, String[] categoryPaths) throws RepositoryException;`	Adds several categories to the given node. @param node: The node to which we add the categories.

Method	Return	Prototype	Description
			@param taxonomyName: The name of the taxonomy. @param categoryPaths: An array of category paths relative to the given taxonomy. @throws RepositoryException: if the categories cannot be added.
addCategories	`void`	`addCategories(Node node, String taxonomyName, String[] categoryPaths, boolean system) throws RepositoryException;`	Adds several categories to the given node. @param node: The node to which we add the categories. @param taxonomyName: The name of the taxonomy. @param categoryPaths: An array of category paths relative to the given taxonomy. @param system: check system provider or not. @throws RepositoryException: if the categories cannot be added.
addCategory	`void`	`addCategory(Node node, String taxonomyName, String categoryPath) throws RepositoryException;`	Adds a new category path to the given node. @param node: the node to which we add the category.

Method	Return	Prototype	Description
			@param taxonomyName: The name of the taxonomy. @param categoryPath: The path of the category relative to the given taxonomy. @throws RepositoryException: if the category cannot be added.
addCategory	`void`	`addCategory(Node node, String taxonomyName, String categoryPath, boolean system) throws RepositoryException;`	Adds a new category path to the given node. @param node: the node to which we add the category. @param taxonomyName: The name of the taxonomy. @param categoryPath: The path of the category relative to the given taxonomy. @param system: check system provider or not. @throws RepositoryException: if the category cannot be added.
getTaxonomyTreeDefaultUserPermission	`Map<String, String[]>`	`getTaxonomyTreeDefaultUserPermission();`	Gets the default permission for the user in taxonomy tree.
addTaxonomyPlugin	`void`	`addTaxonomyPlugin(ComponentPlugin plugin);`	Adds a new taxonomy plugin to the service.

Method	Return	Prototype	Description
			@param plugin: The plugin to adds
init	`void`	`init(String repository) throws Exception;`	Initializes all taxonomy plugins that have been already configured in .xml files. @param repository: The name of repository. @see TaxonomyPlugin. @throws Exception.

| **getCategoryNameLength** | `String` | `getCategoryNameLength();` | Gets the limited length of the category name. |

4.5.2. LinkManager

Supply API to work with the linked node or the link included in a node.

Package org.exoplatform.services.cms.link.LinkManager

Method	Return	Prototype	Description
createLink	`Node`	`createLink(Node parent, String linkType, Node target) throws RepositoryException;`	Creates a new link that is added to the parent node and returns the link. @param parent: The parent node of the link. @param linkType: The primary node type of the link must be a sub-type of exo:symlink, the default value is "exo:symlink" @param target The target of the link. @throws RepositoryException: if

Method	Return	Prototype	Description
			the link cannot be created for any reason.

createLink	`Node`	`createLink(Node parent, Node target)`
		
		`throws RepositoryException;`
			Creates a new node of type exo:symlink, then adds it to the parent node and returns the link node.
			
			@param parent: The parent node of the link to create.
			
			@param target: The target of the link.
			
			@throws RepositoryException: if the link cannot be created for any reason.

createLink	`Node`	`createLink(Node parent, String linkType, Node target, String linkName)`
		
		`throws RepositoryException;`
			Creates a new link that is added to the parent node and returns the link.
			
			@param parent: The parent node of the link.
			
			@param linkType: The primary node type of the link must be a sub-type of exo:symlink, the default value is exo:symlink.
			
			@param target: The target of the link.
			
			@param linkName: The name of the link.
			
			@return
			
			@throws RepositoryException: if

Method	Return	Prototype	Description
			the link cannot be created for any reason.

updateLink	`Node`	`updateLink(Node link, Node target)`
		
		`throws RepositoryException;`
			Updates the target node of the given link.
			
			@param link: The link node to update.
			
			@param target: The new target of the link.
			
			@throws RepositoryException: if the link cannot be updated for any reason.

getTarget	`Node`	`getTarget(Node link, boolean system) throws ItemNotFoundException,`
		
		`RepositoryException;`
			Gets the target node of the given link.
			
			@param link: The node of type exo:symlink.
			
			@param system: Indicates whether the target node must be retrieved using a session system or user session in case we cannot use the same session as the node link because the target and the link are not in the same workspace.
			
			@throws ItemNotFoundException: if the target node cannot be found.
			
			@throws RepositoryException: if an unexpected error occurs while retrieving the target node.

getTarget	`Node`		

Method	Return	Prototype	Description

		`getTarget(Node link)`
		
		`throws ItemNotFoundException, RepositoryException;`
			Gets the target node of the given link using the user.
			
			@param link: The node of type exo:symlink.
			
			@throws ItemNotFoundException: if the target node cannot be found.
			
			@throws RepositoryException: if an unexpected error occurs while retrieving the target node.

isTargetReachable	`boolean`	`isTargetReachable(Node link)`
		
		`throws RepositoryException;`
			Checks if the target node of the given link can be reached using the user session.
			
			@param link: The node of type exo:symlink.
			
			@throws RepositoryException: if an unexpected error occurs .

isTargetReachable	`boolean`	`isTargetReachable(Node link, boolean system)`
		
		`throws RepositoryException;`
			Checks if the target node of the given link can be reached using the user session.
			
			@param link: The node of type exo:symlink.
			
			@param system
			
			@throws RepositoryException if an unexpected error

Method	Return	Prototype	Description
			occurs.

isLink	`boolean`	`isLink(Item item)`
		
		`throws RepositoryException;`
			Indicates whether the given item is a link.
			
			@param item: the item to test.
			
			@return <code>true</code>: if the node is a link, <code>false</code> otherwise.
			
			@throws RepositoryException: if an unexpected error occurs.

getTargetPrimaryNodeType	`String`	`getTargetPrimaryNodeType(Node link)`
		
		`throws RepositoryException;`
			Gives the primary node type of the target.
			
			@param link: The node of type exo:symlink.
			
			@return: the primary node type of the target
			
			@throws RepositoryException: if an unexpected error occurs.

getAllLinks	`List<Node>`	`getAllLinks(Node targetNode, String linkType, String repoName)`
		
		`{}throws Exception;`
			Gives all links of the given node.
			
			@param targetNode: The target node to get links.
			
			@param linkType: The type of link to get.
			
			@param repoName: Name of the repository.

Method	Return	Prototype	Description
			@return: the list of link of the target node with given type. @throw Exception

4.5.3. PublicationManager

PublicationService to manage the publication

Method	Return	Prototype	Description
addLifecycle	`void`	`addLifecycle(ComponentPlugin plugin);`	Adds plugins context for the publication service.
getLifecycle	`Lifecycle`	`Lifecycle getLifecycle(String name) ;`	Gets a specific lifecycle with the given name. @param name: The name of the wanted lifecycle.
getLifecycles	`List<Lifecycle>`	`List<Lifecycle> getLifecycles();`	Gets all the lifecycles which were added to service instances.
getContext	`Context`	`Context getContext(String name) ;`	Gets a specific context with the given names.
getContexts	`List<Context>`	`List<Context> getContexts();`	Gets all the contexts which were added to service instances.
getContents	`List<Node>`	`List<Node> getContents(String fromstate, String tostate, String date, String user, String lang , String workspace) throws Exception;`	Gets all the node. @param fromstate/tostate: the current range state of node. @param user: The last user of node. @param lang: the node's language.

Method	Return	Prototype	Description
			@param workspace: the Workspace of the node's location.
getLifecyclesFromUser	`List<Lyfecycle>`	`List<Lyfecycle> getLifecyclesFromUser(String remoteUser, String state);`	Gets all the Lifecycle of a specific user. @param remoteUser: the current user of publication service. @param state: The current state of the node.

4.5.4. WCMComposer

This class is used to get contents inside the WCM product. You should not access contents directly from the JCR on the front side.

In general, this service stands between publication and cache.

Package org.exoplatform.services.wcm.publication.WCMComposer

Method	Return	Prototype	Description

getContent	`Node`	`getContent(String repository, String workspace, String nodeIdentifier, HashMap<String, String> filters, SessionProvider sessionProvider)`
		
		`throws Exception ;`
			Returns contents at the specified path based on filters.
			
			@param repository: the repository
			
			@param workspace: the workspace
			
			@param path: the path
			
			@param filters: the filters
			
			@param sessionProvider: the session provider
			
			@return a JCR node

Method	Return	Prototype	Description
			@throws Exception: the exception

getContents	`List<Node>`	`getContents(String repository, String workspace, String path, HashMap<String, String> filters, SessionProvider sessionProvider)`
		
		`throws Exception ;`
			Returns contents at the specified path based on filters.
			
			@param repository: the repository
			
			@param workspace: the workspace
			
			@param path: the path
			
			@param filters: the filters
			
			@param sessionProvider: the session provider
			
			@return: a JCR node
			
			@throws Exception: the exception

updateContent	`boolean`	`updateContent(String repository, String workspace, String nodeIdentifier, HashMap<String, String> filters)`
		
		`throws Exception;`
			Updates content.
			
			@param repository: the repository
			
			@param workspace: the workspace
			
			@param path: the path
			
			@param filters: the filters

Method	Return	Prototype	Description
			@return true, if successful.

getAllowedStates	`List<String>`	`getAllowedStates(String mode)`
		
		`throws Exception ;`
			Returns allowed states for a specified mode.
			
			@param mode: the mode
			
			@return a JCR node.
			
			@throws Exception: the exception

cleanTemplates	`void`	`cleanTemplates()`
		
		`throws Exception ;`
			Initializes the templates hashmap.
			
			@throws Exception: the exception

isCached	`boolean`	`isCached()`
		
		`throws Exception;`
			Checks isCache or not.
			
			@return: the state of caches
			
			@throws Exception: the exception

4.5.5. NewFolksonomy

Package org.exoplatform.services.cms.folksonomy.NewFolksonomyService;

Method	Return	Prototype	Description
addPrivateTag	`void`	`addPrivateTag(String[] tagsName, Node documentNode, String repository, String workspace, String userName) throws Exception ;`	Adds a private tag to a document. A folksonomy link will be created in a tag node. @param tagNames: Array of tag name as the children of tree.

Method	Return	Prototype	Description
			@param documentNode: Tags this node by creating a folksonomy link to the node in the tag. @param repository: Repository name @param workspace: Workspace name @param userName: User name @throws Exception
addGroupsTag	`void`	`addGroupsTag(String[] tagsName, Node documentNode,String repository, String workspace, String[] roles) throws Exception ;`	Adds a group tag to a document. A folksonomy link will be created in a tag node. @param tagNames: Array of tag name as the children of tree. @param documentNode: Tags this node by creating a folksonomy link to the node in tag. @param repository: Repository name @param workspace: Workspace name @param roles: User roles @throws Exception
addPublicTag	`void`	`addPublicTag(String treePath, String[] tagsName, Node documentNode, String repository, String`	Adds a public tag to a document. A folksonomy link will be created in a tag node.

Method	Return	Prototype	Description
		`workspace) throws Exception ;`	@param treePath: Path of folksonomy tree @param tagNames: Array of tag name as the children of tree. @param documentNode: Tags this node by creating a folksonomy link to the node in the tag. @param repository: Repository name @param workspace: Workspace name @throws Exception
addSiteTag	`void`	`addSiteTag(String siteName, String[] tagsName, Node node, String repository, String workspace) throws Exception ;`	Adds a site tag to a document. A folksonomy link will be created in a tag node. @param siteName: Portal name @param treePath: Path of folksonomy tree @param tagNames: Array of tag name as the children of tree. @param documentNode: Tags this node by creating a folksonomy link to the node in tag. @param repository: The repository name @param workspace:

Method	Return	Prototype	Description
			Workspace name @throws Exception
getAllPrivateTags	`List<Node>`	`getAllPrivateTags(String userName, String repository, String workspace) throws Exception ;`	Gets all private tags. @param userName: User name @param repository: The repository name @param workspace: Workspace name @return List<Node>
getAllPublicTags	`List<Node>`	`getAllPublicTags(String treePath, String repository, String workspace) throws Exception ;`	Gets all public tags. @param treePath: Folksonomy tree path @param repository: The repository name @param workspace: Workspace name @return List<Node> @throws Exception
getAllGroupTags	`List<Node>`	`getAllGroupTags(String[] roles, String repository, String workspace) throws Exception ;`	Gets all tags by groups. @param roles: Roles of user @param repository: Repository name @param workspace: Workspace name

Method	Return	Prototype	Description
			@return List<Node> @throws Exception
getAllGroupTags	`List<Node>`	`getAllGroupTags(String role, String repository, String workspace) throws Exception ;`	Gets all tags by group. @param role: Role of user @param repository: Repository name @param workspace: Workspace name @return List<Node> @throws Exception
getAllSiteTags	`List<Node>`	`getAllSiteTags(String siteName, String repository, String workspace) throws Exception ;`	Gets all tags of Site. @param siteName: Portal name @param treePath: Folksonomy tree path @param repository: Repository name @param workspace: Workspace name @return List<Node> @throws Exception
getAllDocumentsByTag	`List<Node>`	`getAllDocumentsByTag(String tagPath, String repository, String workspace, SessionProvider sessionProvider)`	Gets all documents which are stored in a tag. @param treeName: The name of folksonomy tree

Method	Return	Prototype	Description
		`throws Exception ;`	@param tagName: The name of a tag @param repository: The repository name @return: a list of documents in atag @throws Exception
getTagStyle	`String`	`getTagStyle(String tagPath, String repository, String workspace) throws Exception ;`	Gets HTML*STYLE*PROP property in styleName node in repository. @param tagPath: Tag path @param workspace: The workspace name @param repository: The repository name @return: The value of property of the styleName node @throws Exception
addTagStyle	`void`	`addTagStyle(String styleName, String tagRange, String htmlStyle, String repository, String workspace) throws Exception ;`	Updates the properties TAG*RATE*PROP and HTML*STYLE*PROP, following the values tagRate, htmlStyle for node in tagPath in repository. @param styleName: Style name @param tagRate: The range of tag numbers

Method	Return	Prototype	Description
			@param htmlStyle: Tag style @param repository: The repository name @param workspace: The workspace name @throws Exception
updateTagStyle	`void`	`updateTagStyle(String styleName, String tagRange, String htmlStyle, String repository, String workspace) throws Exception ;`	Updates the properties TAG*RATE*PROP and HTML*STYLE*PROP, following the value tagRate, htmlStyle for node in tagPath in repository. @param styleName: Style name @param tagRate: The range of tag numbers @param htmlStyle: Tag style @param repository: The repository name @param workspace: The workspace name @throws Exception
getAllTagStyle	`List<Node>`	`getAllTagStyle(String repository, String workspace) throws Exception ;`	Gets all tag style bases of a folksonomy tree. @param repository: The repository name @param workspace:

Method	Return	Prototype	Description
			Workspace name @return List<Node> List tag styles @throws Exception
init	`void`	`init(String repository) throws Exception ;`	Initialize all TagStylePlugin with session in repository name. @param repository: The repository name
removeTagOfDocument	`void`	`removeTagOfDocument(String tagPath, Node document, String repository, String workspace) throws Exception;`	Removes a tag of a given document. @param treeName: The name of a folksonomy tree @param tagName: The name of a tag @param document: The document which is added a link to tagName. @param repository: The repository name @return @throws Exception
removeTag	`void`	`removeTag(String tagPath, String repository, String workspace) throws Exception;`	Removes tag. @param tagPath: The path of the tag @param repository: The repository name

Method	Return	Prototype	Description
			@param workspace: The workspace name
modifyTagName	`Node`	`modifyTagName(String tagPath, String newTagName, String repository, String workspace) throws Exception;`	Modifies the tag name. @param tagPath: The name of the tag @param newTagName: New tag name @param repository: The repository name @param workspace: The workspace name @return @throws Exception
getLinkedTagsOfDocument	`List<Node>`	`getLinkedTagsOfDocument(Node documentNode, String repository, String workspace) throws Exception;`	Gets all tags linked to a given document. @param documentNode: Document node @param repository: The repository name @param workspace: Workspace name @return @throws Exception
getLinkedTagsOfDocumentByScope	`List<Node>`	`getLinkedTagsOfDocumentByScope(int scope, String value, Node documentNode, String repository, String workspace) throws Exception;`	Gets all tags linked to a given document by scope. @param documentNode: Document node

Method	Return	Prototype	Description
			@param repository: The repository name @param workspace: The workspace name @return @throws Exception
removeTagsOfNodeRecursively	void	`removeTagsOfNodeRecursively(Node node, String repository, String workspace, String username, String groups) throws Exception;`	Removes all tags linked to the child nodes of a given node. @param node @param repository @param workspace @throws Exception
addTagPermission	`void`	`addTagPermission(String usersOrGroups);`	Adds given users or groups to tagPermissionList. @param usersOrGroups
removeTagPermission	`void`	`removeTagPermission(String usersOrGroups);`	Removes given users or groups from tagPermissionList. @param usersOrGroups
getTagPermissionList	`List<String>`	`getTagPermissionList();`	Returns tagPermissionList.
canEditTag	`boolean`	`canEditTag(int scope, List<String> memberships);`	Sets the permission to edit a tag for a user. @param scope: Scope @param memberships:

Method	Return	Prototype	Description
			Memberships @return: true if it is possible.
getAllTagNames	`List<String>`	`getAllTagNames(String repository, String workspace, int scope, String value) throws Exception;`	Gets all tag names which start within a given scope. @param repository: Repository @param workspace: Workspace @param scope: scope of tags @param value: value, according to scope, can be understood differently. @return true if it is possible.

4.5.6. ApplicationTemplateManager

This class is used to manage dynamic groovy templates for WCM-based products.

Package org.exoplatform.services.cms.views.ApplicationTemplateManager;

Method	Return	Prototype	Description
addPlugin	`void`	addPlugin(PortletTemplatePlugin portletTemplatePlugin) throws Exception	Adds the plugin. portletTemplatePlugin: the portlet template plugin
getAllManagedPortletNames	`List<String>`	getAllManagedPortletNames(String repository) throws Exception	Retrieves all the portlet names that have dynamic groovy templates managed by service.

Method	Return	Prototype	Description
			repository: the repository @throws Exception the exception
getTemplatesByApplication	`List<Node>`	getTemplatesByApplication(String repository, String portletName, SessionProvider provider) throws Exception;	Retrieves the templates node by application. @param repository: the repository @param portletName: the portlet name @param provider: the provider @return the templates by application @throws Exception: the exception
getTemplatesByCategory	`List<Node>`	getTemplatesByCategory(String repository, String portletName, String category, SessionProvider sessionProvider) throws Exception;	Retrieves the templates node by category. @param repository: the repository @param portletName: the portlet name @param category: the category @param sessionProvider: the session provider @return the templates by category @throws Exception: the exception

Method	Return	Prototype	Description
getTemplateByName	`Node`	getTemplateByName(String repository, String portletName, String category, String templateName, SessionProvider sessionProvider) throws Exception;	Retrieves the template by name. @param repository: the repository @param portletName: the portlet name @param category: the category @param templateName: the template name @param sessionProvider: the session provider @return the template by name @throws Exception: the exception
getTemplateByPath	`Node`	getTemplateByPath(String repository, String templatePath, SessionProvider sessionProvider) throws Exception ;	Gets the template by path. @param repository: the repository @param templatePath: the template path @param sessionProvider: the session provider @return the template by path @throws Exception: the exception
addTemplate	`void`	addTemplate(Node	Adds the template.

Method	Return	Prototype	Description
		portletTemplateHome, PortletTemplateConfig config) throws Exception;	@param portletTemplateHome: the portlet template home @param config: the config @throws Exception: the exception
removeTemplate	`void`	removeTemplate(String repository, String portletName, String catgory, String templateName, SessionProvider sessionProvider) throws Exception;	Removes the template. @param repository: the repository @param portletName: the portlet name @param catgory: the catgory @param templateName: the template name @param sessionProvider: the session provider @throws Exception: the exception

4.5.7. NodeFinder

NodeFinder is used to find a node with a given path. If the path to the node contains sub-paths to exo:symlink nodes, find the real link node.

Method	Return	Prototype	Description
getNode	`Node`	getNode(Node ancestorNode, String relativePath) throws PathNotFoundException, RepositoryException;	Returns the node at relPath related to the ancestor node. @param ancestorNode:

Method	Return	Prototype	Description
			The ancestor of the node to retrieve from which we start. @param relativePath: The relative path of the node to retrieve. @throws PathNotFoundException: If "no", the node exists at the specified path. @throws RepositoryException: if another error occurs.
getNode	`Node`	getNode(Node ancestorNode, String relativePath, boolean giveTarget) throws PathNotFoundException, RepositoryException;	Returns the node at relPath related to the ancestor node. If the node is a link and giveTarget has been set to <code>true</code>, the target node will be returned @param ancestorNode: The ancestor of the node to retrieve from which we start. @param relativePath: The relative path of the node to retrieve. @param giveTarget: Indicates if the target must be returned in case the item is a link @throws PathNotFoundException: If no node exists at the specified path.

Method	Return	Prototype	Description
			@throws RepositoryException: if another error occurs.
getItem	`Item`	getItem(String repository, String workspace, String absPath) throws PathNotFoundException, RepositoryException;	Returns the item at the specified absolute path. @param repository: The name of repository @param workspace: The name of workspace @param absPath: An absolute path @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if another error occurs.
getItemSys	`Item`	getItemSys(String repository, String workspace, String absPath, boolean system) throws PathNotFoundException , RepositoryException;	Returns the item at the specified absolute path. @param repository: The name of repository @param workspace: The name of workspace @param absPath: An absolute path. @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if

Method	Return	Prototype	Description
			another error occurs.
getItem	`Item`	getItem(String repository, String workspace, String : absPath, boolean giveTarget) throws PathNotFoundException, RepositoryException;	Returns the item at the specified absolute path. If the item is a link and giveTarget has been set to <code>true</code>, the target node will be returned. @param repository: The name of repository @param workspace: The name of workspace @param absPath: An absolute path @param giveTarget: Indicates if the target must be returned in case the item is a link @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if another error occurs.
getItemGiveTargetSys	`Item`	getItemGiveTargetSys(String repository, String workspace, String absPath, boolean giveTarget, boolean system) throws PathNotFoundException, RepositoryException;	Returns the item at the specified absolute path. If the item is a link and giveTarget has been set to <code>true</code>, the target node will be returned. @param repository: The name of repository @param workspace: The

Method	Return	Prototype	Description
			name of workspace @param absPath: An absolute path @param giveTarget: Indicates if the target must be returned in case the item is a link @param system: system provider @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if another error occurs.
getItem	`Item`	getItem(Session session, String absPath) throws PathNotFoundException, RepositoryException;	Returns the item at the specified absolute path. @param session: The session is used to get the item @param absPath: An absolute path @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if another error occurs.
getItem	`Item`	getItem(Session session, String absPath, boolean giveTarget) throws PathNotFoundException,	Returns the item at the specified absolute path. If the item is a link and giveTarget has been set to

Method	Return	Prototype	Description
		RepositoryException;	<code>true</code>, the target node will be returned @param session: The session is used to get the item. @param absPath: An absolute path @param giveTarget: Indicates if the target must be returned in case the item is a link @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if another error occurs.
getItemTarget	`Item`	`getItemTarget(Session session, String absPath, boolean giveTarget, boolean system) throws PathNotFoundException, RepositoryException;`	Returns the item at the specified absolute path. If the item is a link and giveTarget has been set to <code>true</code>, the target node will be returned @param session: The session is used to get the item. @param absPath: An absolute path @param giveTarget: Indicates if the target must be returned in case the item is a link.

Method	Return	Prototype	Description
			@param system: system provider @throws PathNotFoundException: if the specified path cannot be found. @throws RepositoryException: if another error occurs.
itemExists	`boolean`	itemExists(Session session, String absPath) throws RepositoryException;	Returns <code>true</code> if an item exists at absPath; otherwise returns <code>false</code> Also returns <code>false</code> if the specified absPath is malformed. @param session: The session is used to get the item. @param absPath: An absolute path @return <code>true</code> if an item exists at absPath; otherwise returns <code>false</code>. @throws RepositoryException: if an error occurs.

4.5.8. JodConverter

JodConverter is used to convert documents into different office formats.

Package org.exoplatform.services.cms.jodconverter.JodConverterService

Method	Return	Prototype	Description
convert	`void`	convert(InputStream input, String formatInput, OutputStream out, String formatOutput) throws Exception;	Converts InputStream in the formatInput format to OutputStream with the formatOutput format. @param input @param formatInput @param out @param formatOutput @throws Exception

4.6. FAQ

4.6.1. How to deploy a workflow?

The ***addPlugin()*** function of **WorkflowServiceContainer** service is used to register a Business Process when a workflow is implemented. Thus, if you want to use a workflow, you are required to configure the workflow service to invoke the ***addPlugin()*** function by adding the ***external-component-plugins*** element to the configuration file.

You have to set values for the name and location of the workflow which you want to use. There are **TWO** ways to configure the location of the workflow.

4.6.1.1. Deploy a workflow inside a .war file

You can use "war:(FOLDER_PATH)" to configure which **.jar** files contain your workflow processes inside the ***.war*** file.

```
<external-component-plugins>
  <target-component>org.exoplatform.services.workflow.WorkflowServiceContainer</target-component>
  <component-plugin>
    <name>deploy.predefined.processes</name>
    <set-method>addPlugin</set-method>
    <type>org.exoplatform.services.workflow.PredefinedProcessesPlugin</type>
    <init-params>
      <object-param>
        <name>predefined.processes</name>
        <description>load of default business processes</description>
        <object type="org.exoplatform.services.workflow.ProcessesConfig">
          <field name="processLocation">
            <string>war:/conf/bp</string>
          </field>
```

```
          <field name="predefinedProcess">
            <collection type="java.util.HashSet">
              <value>
                <string>/exo-ecms-ext-workflow-bp-jbpm-content-2.1.1.jar</string>
              </value>
              <value>
                <string>/exo-ecms-ext-workflow-bp-jbpm-payraise-2.1.1.jar</string>
              </value>
              <value>
                <string>/exo-ecms-ext-workflow-bp-jbpm-holiday-2.1.1.jar</string>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

4.6.1.2. Deploy a workflow inside a .jar file

You can use "classpath:" to configure which **.jar** files contain your workflow processes inside the ***.jar*** file.

```
<external-component-plugins>
  <target-component>org.exoplatform.services.workflow.WorkflowServiceContainer</target-component>
  <component-plugin>
    <name>deploy.predefined.processes</name>
    <set-method>addPlugin</set-method>
    <type>org.exoplatform.services.workflow.PredefinedProcessesPlugin</type>
    <init-params>
      <object-param>
        <name>predefined.processes</name>
        <description>load of default business processes</description>
        <object type="org.exoplatform.services.workflow.ProcessesConfig">
          <field name="processLocation">
            <string>classpath:</string>
          </field>
          <field name="predefinedProcess">
            <collection type="java.util.HashSet">
              <value>
                <string>/exo-ecms-ext-workflow-bp-jbpm-content-myworkflow.jar</string>
              </value>
            </collection>
          </field>
        </object>
      </object-param>
    </init-params>
  </component-plugin>
</external-component-plugins>
```

The notification message is displayed when you deploy a workflow on Jboss. If you use "classpath:" to register, you must put your workflow in the **.jar** files inside the ***gatein.ear/lib*** folder (instead of the **lib** folder) to make it work.