Alfresco Share

Alfresco Share provides a rich web-based collaboration environment for managing documents, wiki content, blogs and more. Share leverages the Alfresco repository to provide content services and utilises the Alfresco Surf Platform to provide the underlying presentation framework.

Contents [hide]

Functionality

The following functions are provided out-of-the-box.

- Site management
- Document libraries
- Wikis
- Blogs
- Discussions
- Calendaring
- Link management
- Site and user dashboards
- Site membership services

More detail on each component is provided in the full feature list.

Customisation

A number of options are available to developers and administrators wishing to customise Alfresco Share to better fit into their environment. Many of these mechanisms are provided by the underlying Surf framework, therefore a knowledge of Surf is considered useful for anyone wishing to implement substantial customisations.

Custom dashlets

Dashlets provide the individual components that together form a dashboard in Alfresco Share. Two types of dashboard are defined out-of-the-box, a *user dashboard* which can be customised on a per-user basis, and a *site dashboard* that is shared between all users of a given site.

Since a dashlet is composed of a single web script, it is easy to add additional definitions by adding web scripts to the`user-dashlet`, `site-dashlet` or `dashlet` (for both types) families in the web-tier. This can be used to expose additional capabilities on Share dashboards, which may display repository data fetched via the RESTful API (or via custom extensions to this) or third party data fetched from XML or JSON sources based on the standard

mechanisms provided by the Surf Platform.

See also: List of Share Custom Dashlets.

Custom actions

Alfresco Share provides a number of actions which users may execute against a particular item stored in a site component, such as *Edit*, *Manage Permissions* and *Delete*, usually found in the component's list view.

The Document Library provides a small framework to allow additional actions to be defined more easily. They may then be added to the Document Library browse view or the Document Details screen with a small amount of configuration.

A guided example of how to add a custom Document Library action is available.

Page components

Alfresco Share uses the Surf page-based model to provide navigation between different areas of the application. Theobject model defines multiple *template instances* that dictate how a pages are 'put together' from their constituent parts, or *components*.

This mechanism allows the behaviour or appearance of individual page components to be modified as required without needing to redefine each page where a component appears.

Since the majority of page components are implemented using the web script runtime, customisations may be implemented via modifications to the out-of-the-box web script files, which can (and should) use the Surf override mechanisms to store these outside of the Share web application where they will be safe from future upgrades, etc.

In addition to overriding the out-of-the-box page components, developers may add additional components within a page by modifying the relevant template instance. Again, an override mechanism allows the new template to be stored in an external location outside of the web application.

Site components

Share sites are initially created with a defined set of *site components*, by default a Document Library, Wiki, Blog, Calendar, Discussions and Links. Site configuration mechanisms allow sufficiently-privileged users to modify this set of components after a site has been created and multiple presets may also be defined via XML configuration, specifying the default set of components for each *site type*.

Out-of-the-box a single Default site preset configuration is defined. This appears as the only option in the type drop-down of the Create Site dialogue.

It is possible for developers to define additional site components that may be added to a site. This requires one or more new template instances plus the page components (most likely Web Scripts) which they will include. When this has been done, the new site component must be wired into the normal set provided by Share, and if required added to the relevant presets for your site types.

When creating new site components it is advisable to copy the structure of an existing component such as the Document Library and modify that, rather than starting from scratch. If what you are attempting to implement involves extending the capabilities of an existing component then you may wish to extend this component rather than duplicating code. This is the approach used by Alfresco in adding records management capabilities to the Document Library for the new Records Library component.

Themes

Out-of-the-box, a single `default` theme is provided in the `themes` directory within the Share webapp, which provides the default CSS and image assets used across all pages. The theme is also explicitly defined via an XML file under the`alfresco/site-data/themes` configuration location.

Additional themes may be defined as follows.

1. Create your new theme directory in the webapp themes directory. You may prefer to create a copy of the default theme and modify this as per your needs.
2. Define the theme with a new XML definition file and place this in the override location `alfresco/web-extension/site-data/theme` outside of the webapp. You may copy the contents of the default.xml file contained in `alfresco/site-data/themes` inside the webapp but you must modify the content of the `<id>` element to match the name of your theme directory. You should also provide an appropriate title for your theme in the `<title>`element.

No user interface is currently provided to allow users or administrators to change the active theme used, but the default theme used across the application may be set by overriding the setting in `web-framework-config-application.xml`. An override file `alfresco/web-extension/web-framework-config-custom.xml` can be defined to allow you to make such changes outside of the webapp as follows.

```
<alfresco-config>

   <!-- Apply custom theme -->
   <config evaluator="string-compare" condition="WebFramework">

      <web-framework>
         <application-defaults>
            <theme>customTheme</theme>
         </application-defaults>
      </web-framework>

    </config>

</alfresco-config>
```

Themes other than the application default may be specified at page execution time by specifying a theme ID as the`theme` URL parameter. For example, to change the theme used on the admin user's user dashboard, use the following URL:

http://localhost:8080/share/page/user/admin/dashboard?theme=default

Active Themes under 3.4

The theming in 3.4 has changed slightly.An override file `alfresco/web-extension/share-config-custom.xml` can be defined to allow you to make such changes outside of the webapp as follows.

```
<alfresco-config>
   <!-- Apply custom theme -->
   <config evaluator="string-compare" condition="WebFramework">
      <web-framework>
         <defaults>
            <theme>customTheme</theme>
         </defaults>
      </web-framework>
    </config>
</alfresco-config>
```

Step by Step custom theme

- Copy an existing themes folder & files, such as /Share/themes/greenTheme to /Share/themes/YOUR_THEME
- Copy /share/WEB-INF/classes/alfresco/site-data/themes/default.xml to /share/WEB-INF/classes/alfresco/site-data/themes/YOUR_THEME.xml
- Edit YOUR_THEME.xml

Change where it mentions â€˜Default Blueâ€™ to a title of your choosing, change â€˜theme.defaultâ€˜ to â€˜theme.YOUR_THEMEâ€˜.

- Edit /YOUR_THEME/presentation.css

Change all occurrences of â€˜greenThemeâ€˜ : .yui-skin-greenTheme for .yui-skin-YOUR_THEME

- Edit /YOUR_THEME/yui/assets/skin.css

Change all occurrences of â€˜greenThemeâ€˜ : .yui-skin-greenTheme for .yui-skin-YOUR_THEME

The above 2 steps can be easily done using a find and replace function in any text editor.

Now select your new theme within Share under the Admin Console.

Now edit the .css files and images to your hearts content, refreshing the browser to see your changes.

Configure UI controls

In Share it is possible to change the following properties to configure the UI behaviour (default values are given in parenthesis):

- Document Details - display web preview (true)
- Username - minimum length (2)
- Password - minimum length (3)
- Search - minimum search term length (1)
- Search - maximum no of results (100)

The default values for these properties are specified in the `share-config.xml` and can be overriden in `alfresco/web-extension/share-config-custom.xml`, which can be created by simply removing the `.sample` suffix from the`alfresco/web-extension/share-config-custom.xml.sample` file. To then override a value start with `share-config.xml`and copy sections as required and paste it into the override file. For example, to override the "minimum search term length" with 0 and extend the "maximum number of search results" to 200, modify the `share-config-custom.xml` as follows:

```
<alfresco-config>
   <config evaluator="string-compare" condition="Search" replace="true">
      <search>
         <!-- override minimum length for search terms with 0 -->
         <min-search-term-length>0</min-search-term-length>
         <!-- override maximum number of results for a search with 200 -->
         <max-search-results>200</max-search-results>
      </search>
   </config>
</alfresco-config>
```

Note 1: You must apply the `replace="true"` attribute to override the entire Search section and not individual properties if several properties exist in the same section.

Note 2: It is important to understand that an individual component still may override the default or custom configured values. When a component instance is defined and bound into the page it is possible to define individual property values in the `<properties>` section of the components xml binding file.

For example, in a standard Share installation the search input field for site members accepts `0` characters to perform the search, even though the default "minimum search term length" is set to `1` in `share-config.xml`.
The `<properties>`section in the component instance xml file (`template.site-members.site-members.xml`) overrides the value:

```
<component>
   <scope>template</scope>
   <region-id>site-members</region-id>
   <source-id>site-members</source-id>
   <url>/components/site-members</url>
   <properties>
      <minSearchTermLength>0</minSearchTermLength>
      <setFocus>true</setFocus>
   </properties>
</component>
```

To override the individual components property simply change the value in `<minSearchTermLength>` by overriding the complete component definition.