Alkacon Software GmbH

Technote

Alkacon OAMP Comments Module

Version: 0.9

Date: Friday, May 23, 2008

1   Table of Content

2   Abstract

This document describes the installation, configuration and usage of the Alkacon OpenCms Add-On Module Package Comments. With the Comments module, it is possible to create highly configurable blog comment function without HTML knowledge.

Once created, configured and published, a comment form may be filled out by website visitors to comment a page.

Note: This module extends the Alkacon OAMP Webform Module, so please also read that module's documentation.

3   General purpose of the Alkacon OAMP Comments Module

The module extends a basic OpenCms installation with the capability to create highly configurable online blog comments. It provides the following features:

- Free definable comment form with and without captcha field.

- Comment moderation option.

- Comment administration in 'direct edit' style.

- Different security levels and user restriction.

4   Installation

Note: To use the Alkacon OAMP Comments module, you need at least OpenCms version 7.0.5. The module is not compatible with older OpenCms versions.

Note: This module extends the Alkacon OAMP Webform Module, so you have to install that module before continuing. Please have in mind that you need at least version 1.2, older versions will not work correctly.

Step by step installation procedure:

1. Go to the OpenCms Administration view

2. Click "Module Management" and select either "Import Module from Server" if the module was placed in the **WEB-INF/packages/modules/** folder of your OpenCms installation, or select "Import Module with HTTP" to upload the module from your local file system

3. Select the Alkacon OAMP Comments module zip file **com.alkacon.opencms.comments_1.0.x.zip** to import

4. Check if the jar file **com.alkacon.opencms.comments.jar** has been deployed in the **WEB-INF/lib/** folder after installation

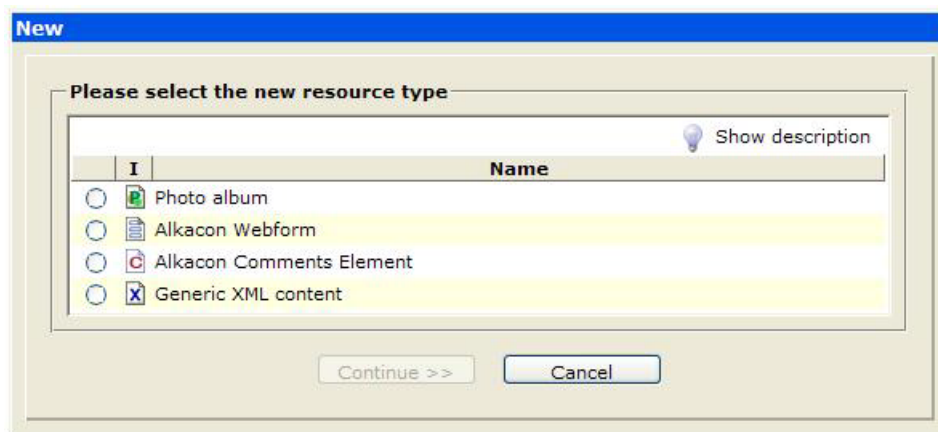5. Restart your servlet container afterwards

5   Module usage

After successful installation of the Comments module, it is ready to use. New comment forms can be set up by creating a new resource of the type "`Alkacon Comments Element`".

5.1 Creating an Alkacon Comments Form

To create a new Comments Form the "new" dialog in OpenCms has to be used:

1.   Click "New" in the Top Bar of the OpenCms Workplace.

2.   In the following dialog click "Structured content" and then "Continue".

3.   Then select "Alkacon Comments Element" and "Continue".

4.   In the following dialog enter the name of the new file.

→ A new file of type "alkacon-comment" will be created and be visible in the Explorer view.

Figure 1: The 2nd stage of the new dialog to create an Alkacon Comment form.

Normally, you will create this kind of files in a separate area of your site, for instance, in a "`/system/`" folder or under the "`_config_`" of template two, or the "`_configuration`" folder of template one. Just be sure the Guest user can read it.

5.2 Editing an Alkacon Comments form

Comment forms are edited by clicking their file symbol in the OpenCms explorer with the right mouse button and choose "edit" from the popup – menu.

The XML content editor for the Comment forms appears.

You will see that the structure of the form is very similar to the one for generic Webforms, some fields has been repositioned and renamed, other few added. One can add as many labelled fields as desired to collect data.

Figure 2: Alkacon OpenCms Comment form editor with comments options.

Here the most important is the Comments Options section, offering following options:

- Moderated:
  If selected the comments have to be approved to be displayed. In this case, the comments can have 3 states:

a. **New**: Will be displayed only to managers until they are approved

b. **Approved**: Will be displayed to everyone

c. **Blocked**: Will be displayed only to managers

Moderators are all users having write permissions on the commented page, these users will have additional options for approving, blocking and deleting comments when viewing the file logged in and in the offline project, like direct edit. Additionally they will also see more details, like for instance, the email address of the writers.

- Comments per page:
  Select the preferred number of comments to display in a page.

- Security Level:
  There are 3 different security levels:

  a. **None**: View and write comments without log in

  b. **Write**: Only view comments without log in, and log in needed for writing.

  c. **View**: Log in needed for viewing and writing comments.

  If security level is view or write, then only users belonging to at least one of the given organizational units or groups are allowed to read or write comments. If no organizational unit and no group are defined, all users are allowed.

- Organizational Units:
  A list of organizational Units

- Groups:
  A list of groups

- Minimized:
  If set, the user gets only a link to open the comments, if not, the comments are immediately displayed.

Be aware, as shown in figure 3, in the notification mail text, that you can use all the field names as macros, and additionally you can use following macros:

- **date**: the current date

- **url**: the page link

Additionally, if using the **com.alkacon.opencms.comments.CmsCommentDynamicFields** class name for the "Dynamic field class" option in the "Advanced Configuration", you can use three different dynamic fields to store some additional information, these are:

- **username**: Login name of the user (if not logged in, Guest)

- **locale**: The locale of the comment

- **ipaddress**: The remote IP Address

Figure 3: User creation options.

5.3 Enabling comments with a given configuration

There are two things to take care of, first to include the comment render element in the right position in your template, and second is to set the '**comments'** property to point to the Comments configuration file to use.

5.3.1   Template adjustment

Just insert the following code in the right place in your template:

```
<!-- begin: #comments -->
<c:set var="comments"><cms:property name="comments" file="search"/></c:set>
<c:if test="${!empty comments}">
      <cms:include
file="%(link.weak:/system/modules/com.alkacon.opencms.comments/elements/comme
nt_main.jsp)" >
            <cms:param name="cmturi" value="${cms.requestContext.uri}" />
      </cms:include>
</c:if>
<!-- end: #comments-->
```

For instance, in template two, a good place would be in file **/system/modules/org.opencms.frontend/templatetwo/templates/main.jsp** close to the end in line 145 before the **<div class='clear'>** stuff. So the comments will appear at the end of the middle column.

5.3.2   Using the right comments configuration file

Additionally you will need to set the '**comments'** property, for single files, or for a folder to enable comments for all files in the folder.

It could be that this property is not yet defined, in which case, you will have to create it first.

And the value of the property has to be a path to the comments configuration file to use, or a configuration file mapping name, for a more user friendly option.

Configuration file mappings are defined as module parameters (in the Administration View > Module Management > Click on the **com.alkacon.opencms.comments** Module > Module Parameters) where you can define parameters having as name: **config:<name>** and as value **<configuration file path>**. Where the <name> is a nice name like '**moderated-webuser'** and the **<configuration file path>** is the path to the configuration file, like '**/system/shared/comments/moderated-webuser.xml'**. So you can use '**moderated-webuser'** instead of '**/system/shared/comments/moderated-webuser.xml'** as '**comments'** property value. A complete module parameter entry will look like:

config:moderated-webuser=system/shared/comments/moderated-webuser.xml

6   Frontend view

Once integrated in your template and configured, you will see something like this:

Figure 4: Demo frontend view for a normal user

Figure 5: Demo frontend view for a manager
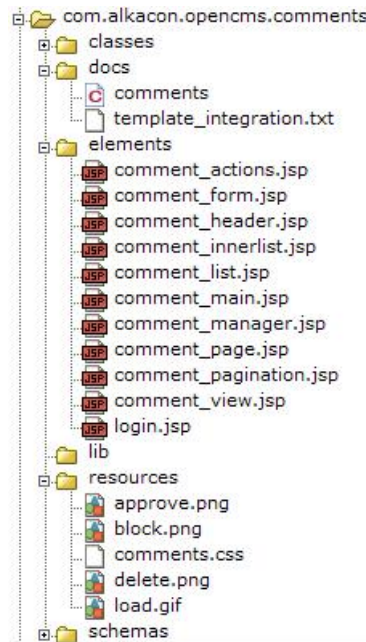
Figure 6: Demo frontend view for the login dialog

Figure 7: Demo frontend view for post dialog

7   Implementation details

The module is composed as described in figure 8:

```
com.alkacon.opencms.comments
    classes
    docs
        comments
        template_integration.txt
    elements
        comment_actions.jsp
        comment_form.jsp
        comment_header.jsp
        comment_innerlist.jsp
        comment_list.jsp
        comment_main.jsp
        comment_manager.jsp
        comment_page.jsp
        comment_pagination.jsp
        comment_view.jsp
        login.jsp
    lib
    resources
        approve.png
        block.png
        comments.css
        delete.png
        load.gif
    schemas
```

Figure 8: Module structure

The **classes** folder contains the resource bundles used for localization. There are two resource bundles:

- **workplace**: for messages displayed in the workplace, mainly the xml content editor.

- **frontend**: for messages displayed in the frontend.

The **docs** folder contains a text file with the piece of code you need to include in your template, and a sample configuration file.

The **elements** folder is the more interesting with all JSP elements:

- **comment_actions.jsp**: executes all administrative actions, like approving, blocking and deleting comments.

- **comment_form.jsp**: this is the post form element

- **comment_header.jsp**: Renders the header of the comment list

- **comment_innerlist.jsp**: Renders the list of comments including pagination

- **comment_list.jsp**: Renders the list of comments including header

- **comment_main.jsp**: This is the main element to include in the template

- **comment_manager.jsp**: This element renders one comment with administration options

- **comment_page.jsp**: Renders a whole page of comments

- **comment_pagination.jsp**: Renders the pagination bar

- **comment_view.jsp**: This element renders a single comment

- **login.jsp**: the login form element

The **lib** folder contains the needed jar file with some few classes.

The **resources** folder contains the used style sheets and icons.

The **schemas** folder contains the **xsd** files defining the configuration xml.