OpenCms Days 2011

Workshop Track:
Creating Plug & Play Modules for OpenCms 8
Rüdiger Kurz
Alkacon Software GmbH
Agenda

- Developing Web-Projects in general
- OpenCms Modules v7 vs. v8
- The module structure
- The module contents
- Creating a new module
There are two approaches to develop Web-Projects in general:

“Tight coupling”
- You can develop a single Module encapsulating everything you need for your Web-Project
- This Module will only work in your concrete project

“Loose coupling”
- You can develop generic Modules realizing a single requirement inside your Web-Project
- Those modules can then be reused in other projects
- Example: One Module for one Content-Type
OpenCms Modules v7 vs. v8

• Version 7 with Template 2 and Web-Form
  – Modification necessary:
    – Adapt the Web-Form-Module itself
    – Adapt your Website-Template
  ➢ Module-Update needs adoption of your Web-Project

• Version 8 with Template 3 and Web-Form
  – Ready to use:
    – Adapting Web-Form not necessary
    – Adapting Website-Template not necessary
  ➢ Module update without any problem
The module structure
Example Scenario

• Imagine you want to create a “plug & play” module that defines only one resource type

• Lets call this resource type “my_content”

• The XSD file for this resource type has the name “content.xsd”

• Then your module should have the following folder structure...
Module Structure

- Module folder
- content model (new)
- sitemap config (new)
- formatters (new)
- schema for the content
The module in detail
The content model typically relies in the module folder under: `config/model`

It is a XML-content instance of the XSD you defined inside your module (`content.xsd`)

This model is used as the default content if you create a new resource of the type `my_content` with the container page editor

It is referenced by the sitemap `config`
The sitemap configuration is a new resource type shipped with OpenCms 8 by default. The main function of that file is to define the resource types that are available in the container page editor. Typically this file is stored below the Site e.g. /sites/default/_config/sitemap.config.

For creating a “plug & play” module we put a sitemap configuration in the module. The path to this sitemap configuration file has to be configured as module parameter (config.sitemap):

config.sitemap=/system/modules/.../sitemap.config
Sitemap Configuration

- Sitemap configurations can rely inside the Module or inside a Sitemap

- When the "container page editor" tries to determine which configuration to take, it looks inside the module and in the Sitemap

- That makes it possible to configure a default behavior for a fresh module installation

- If the "container page editor" finds a configuration on both locations the Sitemap is stronger
Sitemap Configuration

The field: "Source file" points on the content model. That tells the container page editor which content to use as default.

The field: "Pattern" defines the file pattern. That tells the container page editor which file name to use for new contents.
• The formatter’s concept is the key technique for creating “plug & play” modules
• Formatters can be configured for each resource-type
• This is typically done in the XSD
• The new configuration allows to specify attributes that tell the formatter into which container it fits:
  – **uri**: The path to the formatter JSP
  – **type**: specifies the type of the container the formatter is compatible with
  – **minwidth**: The minimum width the container must have to hold the formatter
  – **maxwidth**: The maximum width the container must have to hold the formatter
• Define a formatter:

```xml
[…]
  <xsd:annotation>
    <xsd:appinfo>
      […]
    </xsd:appinfo>
    <formatters>
      <formatter uri="/path/to/JSP"
                   type="*"
                   minWidth="100"
                   maxWidth="500" />
    </formatters>
  </xsd:appinfo>
</xsd:annotation>
[…]
```
Formatters as Key-Technique

- By setting the `type` to "*" the formatter can be used for each container.
- The attributes `minwidth` and `maxwidth` restrict this formatter to be used in containers whose width is in-between.

- A JSP that uses the `<cms:formatter>`-tag can access the configured width of the container.

- ... and can then change the presentation of a content dynamically.

- That makes it possible to create formatters without any knowledge of the template.
<%@page buffer="none" session="false" taglibs="c,cms"%>
<cms:formatter var="content" val="value">
  <div class="view-article">
    <h2>${value.Title}</h2>
    <div class="paragraph">
      <c:set var="imgwidth"/>
      ${((cms.container.width) / 2) - 25}
      </c:set>
      <cms:img src="${value.Image}" width="${imgwidth}"/>
      ${value.Text}
    </div>
  </div>
</cms:formatter>
In spite of the power of formatters it can happen that you use a third party module that won't work with your template.

In that case it is possible to write and configure your own formatter without changing the third party module.

**Advantage**: You can update the third party module without changes.
Create a new module
Creating a new module

• Create a module with the OpenCms workplace as usual

• Create the folder structure inside the module

• Create a XSD schema for the content you want to offer with your module

• Configure the resource type and the explorer type as usual in the opencms-modules.xml

• Create a default content for the newly created schema
Creating a new module

- Create a formatter that renders the content
- Add the formatter to the XSD of your content
- Create a sitemap configuration file with the help of the new wizard of OpenCms
- Edit the Sitemap configuration and set in the field “Source file” the path to your default content inside the module
- Set in the field “Pattern” the pattern you want to have for new files (c_%(number).html)
- Create a module parameter:
  config.sitemap=/system/modules/.../your.sitemap.config
Questions

¿Preguntas?

Questiones?

Fragen?
Thank you very much for your attention

Rüdiger Kurz
Alkacon Software

http://www.alkacon.com
http://www.opencms.org