Alkacon Software GmbH

Technote

Alkacon OAMP Web Optimization Module

Version: 0.9.0

Date: 24.03.2009

1   Table of Content

2 Abstract

This document describes the installation, configuration and usage of the Alkacon OpenCms Add-On Module Package Web Optimization. With the Web Optimization module, it is possible to minify, obsfuscate and merge css and js files using YUI Compressor, as well as create on-the-fly image sprites.

3 General purpose of the Alkacon OAMP Web Optimization Module

The module extends a basic OpenCms installation with the capability to improve the performance of your webpages by providing:

- css and js files compression, resulting in less download times.

- css and js files merging, to reduce the number of HTTP requests.

- css image sprites, to reduce the image weight as well as the number of HTTP requests.

4 Installation

Note: Despite it has not been tested with older OpenCms versions, the Alkacon OAMP Web Optimization module should be compatible with all OpenCms versions starting with 6.2.

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 Web Optimization module zip file `com.alkacon.opencms.weboptimization_x.x.x.zip` to import

4. Check if the jar file `com.alkacon.opencms.weboptimization.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 OAMP Web Optimization module, it is ready to use. And you will have 3 new resource types, you can create using the "**New File Wizard**" on the "**structured content**" page:

- Alkacon Optimized JS Resource

- Alkacon Optimized CSS Resource

- Alkacon Sprite Resource

5.1 Creating a new optimized resource

To create a new optimized file 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 the optimized resource type you want to create and "**Continue**".

4. In the following dialog enter the name of the new file. Be aware that the file extension to optimized JS resources **HAS** to be `js`, the as for optimized CSS resources **HAS** to be `css`. For sprites, you can use any by Java supported image file extension, but it is recommended to use the `png` extension.

→ A new file of type "`alkacon-optimized-js`", "`alkacon-optimized-css`" or "`alkacon-sprite`" will be created and be visible in the Explorer view.
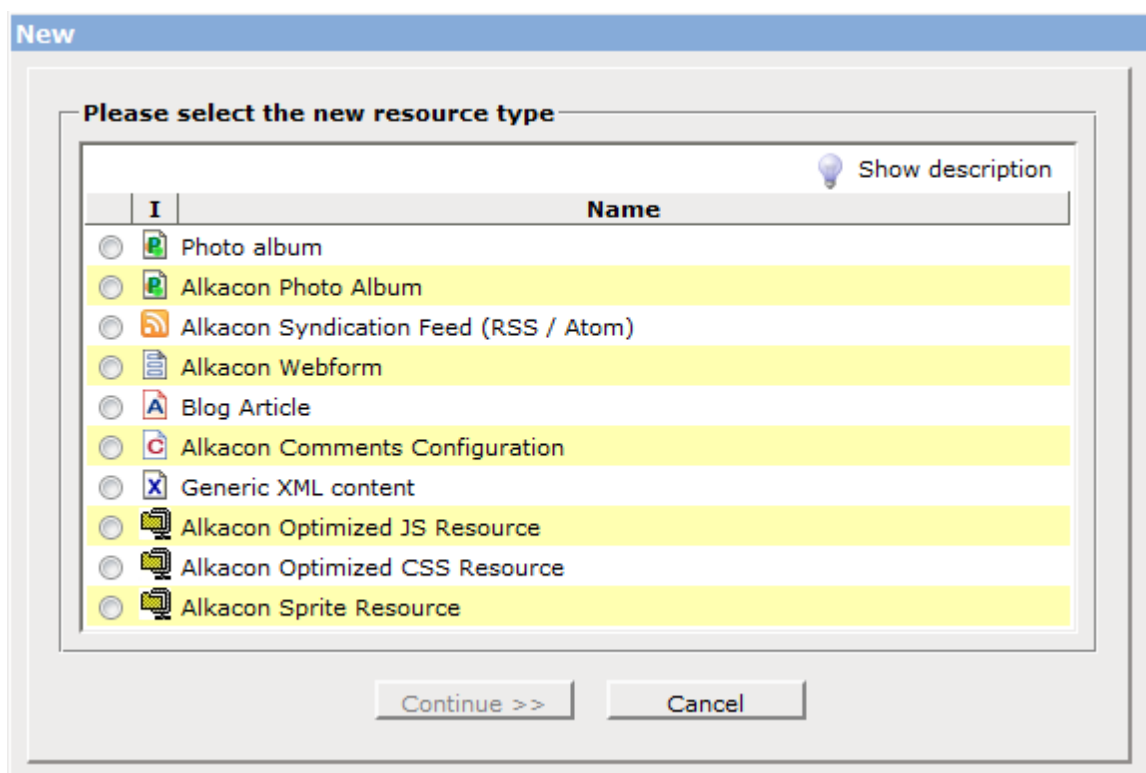

Figure 1: The 2nd stage of the new dialog to create an Alkacon Optimized Resource.

A good starting point is to check the sample files:

`/system/modules/com.alkacon.opencms.weboptimization/example`

5.2 Editing an Alkacon Optimized File

Optimized files 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 optimized resource appears and you mainly have to add the resources you want to optimize.
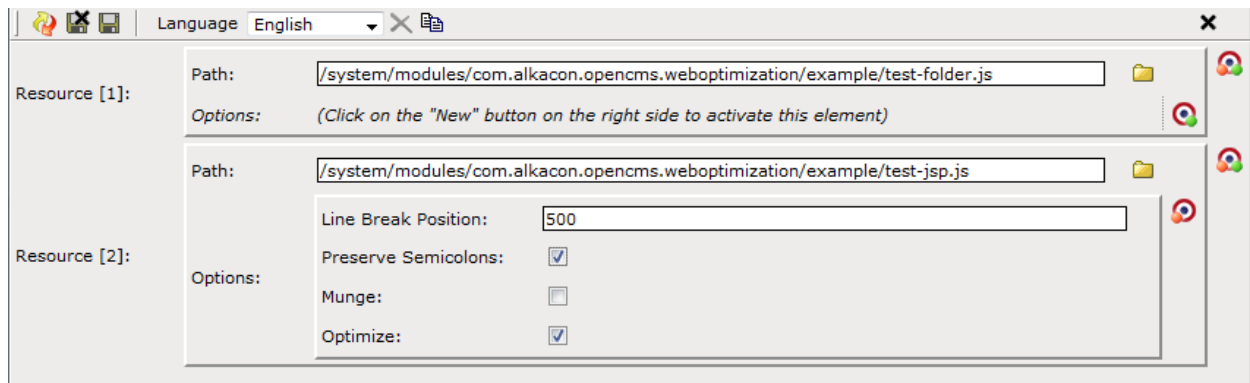
Figure 2: Alkacon Optimized JS Resource editor with optimization options.

For optimized js or css resources, the resource can also be a folder, in which case all files in it with the same extension will be merged. You can also include other optimized resources.

Additionally, for optimized js or css resources only, you can specify the optimization options. For this refer to the YUI Compressor documentation: http://www.julienlecomte.net/yuicompressor/README
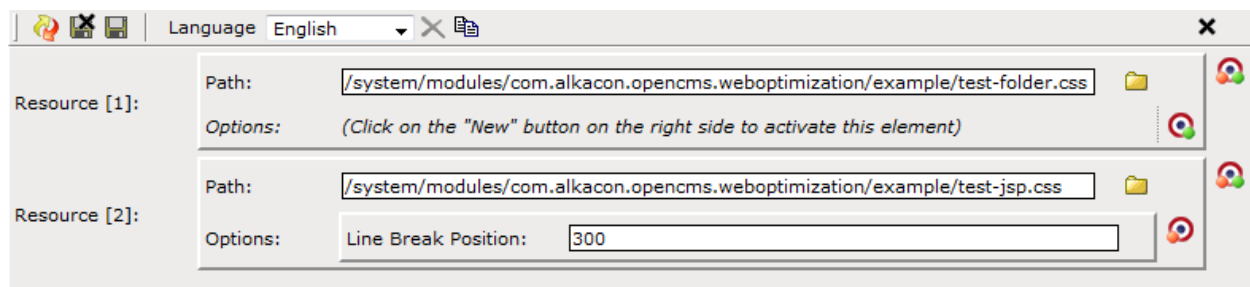

Figure 3: Alkacon Optimized CSS Resource editor with optimization options.

For sprite resources, you have to specify the position where to insert the image into the sprite, and additionally a css selector in case you use the automatic css rules writer, see section 8.
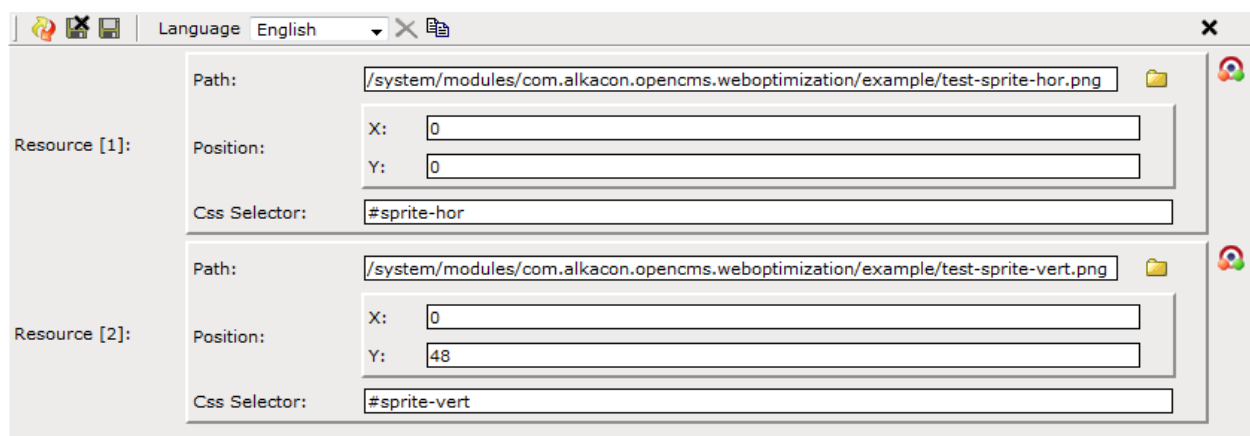
You can also include other sprite resources.

Figure 4: Alkacon Sprite Resource editor.

6 Implementation details
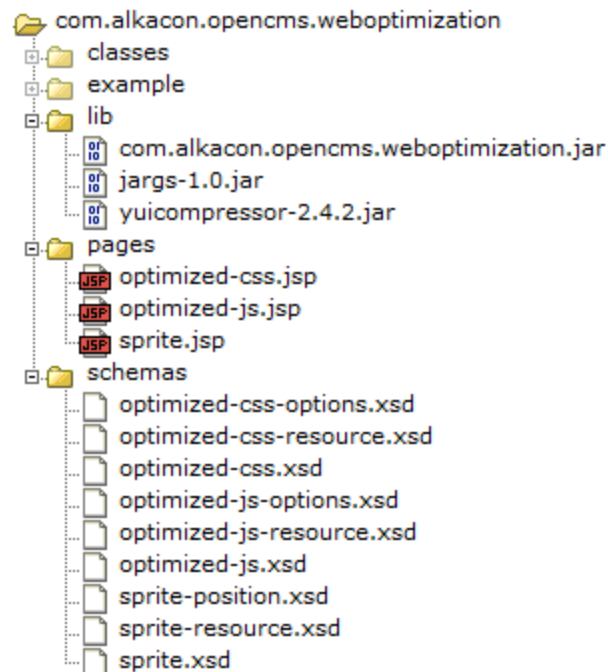
The module is composed as described in figure 3:


```
com.alkacon.opencms.weboptimization
    classes
    example
    lib
        com.alkacon.opencms.weboptimization.jar
        jargs-1.0.jar
        yuicompressor-2.4.2.jar
    pages
        optimized-css.jsp
        optimized-js.jsp
        sprite.jsp
    schemas
        optimized-css-options.xsd
        optimized-css-resource.xsd
        optimized-css.xsd
        optimized-js-options.xsd
        optimized-js-resource.xsd
        optimized-js.xsd
        sprite-position.xsd
        sprite-resource.xsd
        sprite.xsd
```

Figure 5: Module structure

The `classes` folder contains the resource bundles used for localization. There is one resource bundle `workplace` for messages displayed in the workplace, mainly the xml content editor.

The `example` folder contains some sample optimized files.

The `lib` folder contains the needed jar file with some few classes, as well as the need YUI libraries.

The `pages` folder contains the jsp files used to actually merge and optimize the files.

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

7 Performance

Since the target can be any OpenCms resource, in special, also jsp files. You have to be aware that jsp has to be actually executed.

This is true at least for the first request, later requests will be handled by the FlexCache, by default, in the Online project at least.

It is also recommended to static export the optimized file for optimal performance, using the Export On Demand feature,but be aware that in this case you have to touch the optimized file every time you change an included file so the optimized file is exported again with the changes, unless, of course, you set up a special export rule.

8 Additional functionality

This module also provides helper methods for including your optimized files in your templates in the best and easiest way.

For optimized JS resources, this code:

```
<%@ page session="false" import="com.alkacon.opencms.weboptimization.*" %><%
  CmsOptimizationJs oJs = new CmsOptimizationJs(pageContext, request, response);
  oJs.includeDefault("%(link.strong:optimized.js)");
%>
```

Will generate this (in the Online Project):

```
<script type="text/javascript" src="optimized.js" ></script>
```

Or this (working offline):

```
<script type="text/javascript" src="original1.js" ></script>
<script type="text/javascript" src="original2.js" ></script>
```

For optimized CSS resources, this code:

```
<%@ page session="false" import="com.alkacon.opencms.weboptimization.*" %><%
  CmsOptimizationCss oCss = new CmsOptimizationCss(pageContext, request, response);
  oCss.includeDefault("%(link.strong:optimized.js)");
%>
```

Will generate this (in the Online Project):

```
<link rel="stylesheet" media="all" type="text/css" href="optimized.css" />
```

Or this (working offline):

```
<link rel="stylesheet" media="all" type="text/css" href="original1.css" />
<link rel="stylesheet" media="all" type="text/css" href="original2.css" />
```

And for sprite resources, this code:

```
<%@ page session="false" import="com.alkacon.opencms.weboptimization.*" %><%
  CmsOptimizationSprite oS =
                new CmsOptimizationSprite(pageContext, request, response);
  oS.includeDefault("%(link.strong:sprite.png)");
%>
```

Will generate this (in the Online Project):

```
sel1 { background-image: url(optimized.png); background-position: -Xpx -Ypx; }
sel2 { background-image: url(optimized.png); background-position: -XXpx -YYpx; }
```

Or this (working offline):

```
sel1 { background-image: url(original1.png); }
sel2 { background-image: url(original2.png); }
```

Of course, there are also methods to force the optimized or the original versions, namely **includeOriginal()** and **includeOptimized()**.