Automated Plugin Loading PHP Documentation

This page provides information on how a JSP script can be used to automatically load plugins into EditLive!. For more information on plugins, see the Creating and Using Plugins in the Applet article in the Developer Guide for this SDK.

Getting Started

Required License

EditLive!'s Advanced API and Plugin functionality is only supported if an EditLive! Enterprise Edition license has been installed for the editor or if the user is still within their 30 day trial period.

Required Skills

The following skills are required prior to working with this sample:

- Creating reusable functions in Java Server Pages
- Basic client-side JavaScript

Set Up Your Server
Ensure you have set up your Web server for Java server-side processing, as described in the EditLive! Install Guide.

Overview

In this example, EditLive! is created using the Javascript Load Time Methods. A function is called in a JSP script which will write out additional load-time properties to instantiate each plugin located in a specific directory.

A database is not required for this example. You can not perform any saving of document content in this example due to the lack of server-side processing in the example code and the absence of a database.

This sample demonstrates how to perform the following with EditLive! and JSP:

- Iterate through all plugins located in a specific directory and dynamically call to the EditLive! load-time properties to load each plugin.

JSP Script for Automatically Adding Plugins

pluginLoader.jsp

The pluginLoader.jsp script contains a Java function called loadPlugins. loadPlugins contains a routine for searching through a specified directory, locating any instances of .xml files. If an .xml file is found, a call to the addPluginAsText property for EditLive! is written to the page.

loadPlugins requires the following parameters:

- **out** - The JspWriter instance used by the ServletResponse to write to the webpage.
- **elName** - The name of the javascript variable used to specify load-time and run-time properties of EditLive!
- **pluginsDirectory** - The path to the directory on the server containing all of the desired EditLive! plugins.
- **pluginsURL** - The URL for the directory on the server containing all of the desired plugins.

Initializing EditLive! for Java and Automatically Loading the Plugins

example.jsp

To embed EditLive! within a Web page and automatically load all plugins found in a specified directory, several steps are required. Each of these steps is explained here with code samples provided.

1. Create an instance of EditLive! using the javascript Load Time Methods.

```
<html>
       <head>
               <title>Automated Plugin Loading Example - JSP</title>
               <link rel="stylesheet" href="stylesheet.css">
               <!--
               Include the EditLive! JavaScript Library
               -->
               <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
       </head>
       <body>

               <h1>Automated Plugin Loading Example</h1>

               <p>This example depicts how a JSP script can be used to add all of the plugins located
               in a specific to an EditLive! instance.</p>

               <!--
               The instance of EditLive!
               -->
               <script language="JavaScript">
                       // Create a new EditLive! instance with the name "ELApplet", a height of 400 pixels and
a width of 700 pixels.
                       var editlive = new EditLiveJava("ELApplet", 700, 400);

                       // This sets a relative or absolute path to the XML configuration file to use
                       editlive.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");

                       // .show is the final call and instructs the JavaScript library (editlivejava.js) to
insert a new EditLive! instance
                       //  at the this location.
                       editlive.show();
               </script>
       </body>
</html>
```

2. Define the Content type for the page and created a reference to the pluginLoader.jsp script.

```
<%@page contentType="text/html"%>

<%@ include file="pluginLoader.jsp" %>

<html>
       <head>
               <title>Automated Plugin Loading Example - JSP</title>
               <link rel="stylesheet" href="stylesheet.css">
               <!--
               Include the EditLive! JavaScript Library
               -->
               <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
       </head>
       <body>

               <h1>Automated Plugin Loading Example</h1>

               <p>This example depicts how a JSP script can be used to add all of the plugins located
               in a specific to an EditLive! instance.</p>

               <!--
               The instance of EditLive!
               -->
               <script language="JavaScript">
                       // Create a new EditLive! instance with the name "ELApplet", a height of 400 pixels and
a width of 700 pixels.
                       var editlive = new EditLiveJava("ELApplet", 700, 400);

                       // This sets a relative or absolute path to the XML configuration file to use
                       editlive.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");

                       // .show is the final call and instructs the JavaScript library (editlivejava.js) to
insert a new EditLive! instance
                       //  at the this location.
                       editlive.show();
               </script>
       </body>
</html>
```

3. Create the Strings representing the path and the URL to the plugins directory on the server.

```
<%@page contentType="text/html"%>

<%@ include file="pluginLoader.jsp" %>

<html>
       <head>
               <title>Automated Plugin Loading Example - JSP</title>
               <link rel="stylesheet" href="stylesheet.css">
               <!--
               Include the EditLive! JavaScript Library
               -->
               <script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
       </head>
       <body>

               <h1>Automated Plugin Loading Example</h1>

               <p>This example depicts how a JSP script can be used to add all of the plugins located
               in a specific to an EditLive! instance.</p>

               <!--
               The instance of EditLive!
               -->
               <script language="JavaScript">
                       // Create a new EditLive! instance with the name "ELApplet", a height of 400 pixels and
a width of 700 pixels.
                       var editlive = new EditLiveJava("ELApplet", 700, 400);

                       // This sets a relative or absolute path to the XML configuration file to use
                       editlive.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");

                       <%
                               String pluginsDir = getServletContext().getRealPath("/examplePlugins/");
                               String pluginsURL = request.getScheme() + "://" + request.getServerName() + ":"
+ request.getServerPort() + request.getContextPath() + "/examplePlugins/";
                       %>

                       // .show is the final call and instructs the JavaScript library (editlivejava.js) to
insert a new EditLive! instance
                       //  at the this location.
                       editlive.show();
               </script>
       </body>
</html>
```

4. Call to the loadPlugins function (located in the pluginLoader.jsp script), passing the required parameters.

```
<%@page contentType="text/html"%>

<%@ include file="pluginLoader.jsp" %>

<html>
	<head>
		<title>Automated Plugin Loading Example - JSP</title>
		<link rel="stylesheet" href="stylesheet.css">
		<!--
		Include the EditLive! JavaScript Library
		-->
		<script src="../../redistributables/editlivejava/editlivejava.js" language="JavaScript"></script>
	</head>
	<body>

		<h1>Automated Plugin Loading Example</h1>

		<p>This example depicts how a JSP script can be used to add all of the plugins located
		in a specific to an EditLive! instance.</p>

		<!--
		The instance of EditLive!
		-->
		<script language="JavaScript">
			// Create a new EditLive! instance with the name "ELApplet", a height of 400 pixels and
a width of 700 pixels.
			var editlive = new EditLiveJava("ELApplet", 700, 400);

			// This sets a relative or absolute path to the XML configuration file to use
			editlive.setConfigurationFile("../../redistributables/editlivejava/sample_eljconfig.
xml");

			<%
				String pluginsDir = getServletContext().getRealPath("/examplePlugins/");
				String pluginsURL = request.getScheme() + "://" + request.getServerName() + ":"
+ request.getServerPort() + request.getContextPath() + "/examplePlugins/";

				loadPlugins(out, "editlive", pluginsDir, pluginsURL);
			%>

			// .show is the final call and instructs the JavaScript library (editlivejava.js) to
insert a new EditLive! instance
			//  at the this location.
			editlive.show();
		</script>
	</body>
</html>
```

Summary

Using a server-side script, developers can store all desired plugins in a single directory and ensure each plugin is loaded with EditLive!. This solution can cut down on maintenance times required for continually hand-coding load-time properties for each new plugin desired to function with EditLive!.