

# Ckeditor Plugin - Reference Documentation

Stefano Gualdi

Version 4.5.4.0

# Table of Contents

1. Introduction .....	1
1.1. License .....	1
1.2. Current version.....	1
1.3. Source code .....	1
2. Getting started .....	2
2.1. Installation.....	2
2.2. Usage .....	2
3. Configuration .....	3
3.1. Sample config .....	3
4. Tags .....	5
4.1. resources .....	5
4.2. config.....	5
4.3. editor .....	6
4.4. fileBrowser.....	8
4.5. fileBrowserLink .....	9
5. Security .....	10
6. Advanced topics .....	12
6.1. Custom upload types .....	12
6.2. Internal URI's customization.....	12
7. History .....	13

# Chapter 1. Introduction

Welcome to the CKEditor plugin for Grails.

This plugin implements the integration layer between Grails and the [CKEditor](#) web rich text editor.

This plugin can be used to integrate rich web editing functionality in Grails applications.

The plugin is written in Groovy/Grails without any dependencies on external libraries.

## 1.1. License

This plugin is released under the [Apache License, Version 2.0](#)

Please refer to <http://ckeditor.com/license> for the licensing schemes provided by [CKSource](#)

## 1.2. Current version



The versioning model has changed. Since version 3.3.2 the version number of the plugin will reflect the one of the underline integrated CKEditor library. If necessary a 4th level point release number will be used for successive changes on the plugin's code with same version of CKEditor.

Eg.

CKEditor release	Initial plugin version	Maintenance release
3.3.2	3.3.2.0	3.3.2.1
3.4	3.4.0	3.4.0.1
3.6.6.1	3.6.6.1.0	3.6.6.1.1

Current version is **4.5.4.0**

Based on [CKEditor](#) version **4.5.4**

This version is developed with **Grails 3.1.4**

## 1.3. Source code

The full source code for this plugin can be found on [GitHub](#)

For issues, improvements or new features go to the plugin's [JIRA page](#)

To contact me directly my email address is stefano DOT gualdi AT gmail DOT com

Also, feel free to send me any correction about this document.

# Chapter 2. Getting started

## 2.1. Installation

To install the plugin add a dependency to BuildConfig.groovy:

```
plugins {  
    compile ":ckeditor:4.5.4.1"  
}
```

To install the plugin with Grails 3.x:

```
repositories {  
    maven { url "http://dl.bintray.com/stefanogualdi/plugins" }  
}  
dependencies {  
    compile "org.grails.plugins:ckeditor:4.5.4.0"  
}
```

You can find the dependency declaration for the latest version on the [plugin portal page](#).

## 2.2. Usage

To start using the plugin simply add the following code to the page's head:

```
<head>  
...  
    <ckeditor:resources/>  
...  
</head>
```

and in the place you want to render the editor:

```
<ckeditor:editor name="myeditor" height="400px" width="80%">  
    ${initialValue}  
</ckeditor:editor>
```

Please refer to the [Configuration](#) section to understand how to configure the file manager.

# Chapter 3. Configuration

The plugin can be configured with the standard Config.groovy file.

Actually are available the following options:

Config key	Default	Description
<b>ckeditor.config</b>	none	File name, relative to the webapp root, for the custom config file
<b>ckeditor.skipAllowedItemsCheck</b>	false	Skip the check on valid options names
<b>ckeditor.defaultFileBrowser</b>	none	Define the default file browser if not defined in tags. Possible values are: standard, ofm
<b>ckeditor.upload.basedir</b>	/uploads /	Base directory relative to webapp root if baseurl not defined, otherwise the absolute path where to store the uploaded files
<b>ckeditor.upload.baseurl</b>	none	The base URL to access the uploaded files
<b>ckeditor.upload.enableContentController</b>	false	Automatically create a mapping to view files when using baseurl setting
<b>ckeditor.upload.overwrite</b>	false	Overwrite files on upload
<b>ckeditor.upload.link.browser</b>	false	Enable file browser for file objects
<b>ckeditor.upload.link.upload</b>	false	Enable upload tab for file objects
<b>ckeditor.upload.link.allowed</b>	[ ]	Extensions allowed for file objects
<b>ckeditor.upload.link.denied</b>	[ ]	Extensions denied for file objects
<b>ckeditor.upload.image.browser</b>	false	Enable file browser for image objects
<b>ckeditor.upload.image.upload</b>	false	Enable upload tab for image objects
<b>ckeditor.upload.image.allowed</b>	[ ]	Extensions allowed for image objects
<b>ckeditor.upload.image.denied</b>	[ ]	Extensions denied for image objects
<b>ckeditor.upload.flash.browser</b>	false	Enable file browser for flash objects
<b>ckeditor.upload.flash.upload</b>	false	Enable upload tab for flash objects
<b>ckeditor.upload.flash.allowed</b>	[ ]	Extensions allowed for flash objects
<b>ckeditor.upload.flash.denied</b>	[ ]	Extensions denied for flash objects

## 3.1. Sample config

```

ckeditor {
  config = "/js/myckconfig.js"
  skipAllowedItemsCheck = false
  defaultFileBrowser = "ofm"
  upload {
    basedir = "/uploads/"
    overwrite = false
    link {
      browser = true
      upload = false
      allowed = []
      denied = ['html', 'htm', 'php', 'php2', 'php3', 'php4', 'php5',
        'phtml', 'phtml', 'phtml', 'inc', 'asp', 'aspx', 'ascx', 'jsp',
        'cfm', 'cfc', 'pl', 'bat', 'exe', 'com', 'dll', 'vbs', 'js',
        'reg',
        'cgi', 'htaccess', 'asis', 'sh', 'shtml', 'shtm', 'phtm']
    }
    image {
      browser = true
      upload = true
      allowed = ['jpg', 'gif', 'jpeg', 'png']
      denied = []
    }
    flash {
      browser = false
      upload = false
      allowed = ['swf']
      denied = []
    }
  }
}
}

```



For security reasons it is important to define allowed/denied extensions.

# Chapter 4. Tags

The plugin defines the following custom tags:

Tag	Description
<a href="#">resources</a>	Creates the necessary import for the CKEditor's javascripts files
<a href="#">config</a>	Configure the editor instance
<a href="#">editor</a>	Renders the wysiwyg editor
<a href="#">fileBrowser</a>	Creates a link to open just the file browser
<a href="#">fileBrowserLink</a>	Creates an URL to a file browser

all the tags belongs to the namespace **ckeditor**

## 4.1. resources

This tag renders the javascript ref required to use the editor.

Must be used inside the head tag of the page in which you want to use the editor.

```
<html>
  <head>
    <ckeditor:resources />
  </head>

  <body>
  </body>
</html>
```

By default the minified ckeditor's version will be loaded. Adding `minified="false"` will load the not minified, version. This can be useful for debugging purposes.

During the war generation the source will be stripped and the minified code will always be loaded.

## 4.2. config

This tag give full access to every configuration option exposed by CKEditor.

The config tag provides two syntax for simple and complex option definition.

All the configuration options defined with this tag will be used in every editor instance defined after it.

Option names are case sensitive and must be written as indicated in the official documentation.

For a full reference of the available options please refer to the [official documentation](#)

### 4.2.1. Simple option

```
<ckeditor:config OptionName="option_value" />
```

Many options can be set in the same tag:

```
<ckeditor:config  
  OptionOne="option_one_value"  
  OptionTwo="option_two_value" />
```

Examples:

Set height and width for all the following editor's instances

```
<ckeditor:config  
  height="300px"  
  width="50%" />
```

Set the enter key handling strategy for all the following editor's instances

```
<ckeditor:config enterMode="CKEDITOR.ENTER_DIV" />
```

### 4.2.2. Complex option

```
<ckeditor:config var="option_name">  
[structured_option_value]  
</ckeditor:config>
```

Examples:

Define a new toolbar

```
<ckeditor:config var="toolbar_Mytoolbar">  
[  
  [ 'Source', '-', 'Bold', 'Italic' ]  
]  
</ckeditor:config>
```

## 4.3. editor

This tag renders the wysiwyg editor in the page



```
<ckeditor:editor name="text">
Initial text
</ckeditor:editor>
```

available attributes are:

Attribute	Description
<b>id</b>	Id of the instance to be created. If not specified defaults to <b>name</b>
<b>name</b>	Field name (default: "editor")
<b>userSpace</b>	Name of the file user space to use for file browsing and uploads (default: "")
<b>fileBrowser</b>	File manager to use. Possible values: ofm, standard (default: "ofm")
<b>viewMode</b>	View mode for Open File Manager. Possible values: grid, list (default: "grid")
<b>showThumbs</b>	Show images' thumbnails in Open File Manager. Possible values: true, false (default: false)
<b>removeInstance</b>	Remove ckeditor instance before creation. Useful for ajax forms. Possible values: true, false (default: false)
<b>options</b>	Configuration options, as defined in the config tag section.

in the editor tag you can also specify every configuration option you like. Those options will be local to the instance that defines them.

Examples:

Renders an editor with custom size and a simple toolbar

```
<ckeditor:editor name="myeditor" height="400px" width="80%" toolbar="Basic">
${initialValue}
</ckeditor:editor>
```

Renders an editor with a custom toolbar defined in the page

```
<ckeditor:config var="toolbar_Mytoolbar">
[
  [ 'Source', '-', 'Bold', 'Italic' ]
]
</ckeditor:config>

<ckeditor:editor name="myeditor" toolbar="Mytoolbar">
${initialValue}
</ckeditor:editor>
```



For security reasons the **userSpace** attribute cannot contain any special character. Any invalid character will be removed.

## 4.4. fileBrowser

Renders a link to open a stand alone file browser not associated with an editor instance

```
<ckeditor:fileBrowser>Open file browser</ckeditor:fileBrowser>
```

available attributes are:

Attribute	Description
<b>type</b>	Type of resource (default: "")
<b>userSpace</b>	Name of the file user space to use for file browsing and uploads (default: "")
<b>fileBrowser</b>	File manager to use. Possible values: ofm, standard (default: "standard")
<b>viewMode</b>	View mode for Open File Manager. Possible values: grid, list (default: "grid")
<b>showThumbs</b>	Show images' thumbnails in Open File Manager. Possible values: true, false (default: false)
<b>target</b>	Target attribute for the generated <A> tag (default: "")

if you just need the link to the file browser there is a **fileBrowserLink** tag:

```
<a href="{ckeditor.fileBrowserLink(type:'Image', userSpace:'userone')}}">Open file browser</a>
```

If you want get back the path of the selected item in the file browser simply define a javascript function called **ckeditorFileBrowserItemSelected** in the page containing the opening link.

```
<script type="text/javascript" charset="utf-8">
    function ckeditorFileBrowserItemSelected(path) {
        // do whatever you want with path
        alert(path);
    }
</script>
```

Keep in mind that the stand alone file browser is designed to be opened in a separate browser window.

This option is only valid when using Open File Manager.

## 4.5. fileBrowserLink

See [fileBrowser](#) tag

# Chapter 5. Security

The plugin defines some url mappings to the controllers implementing the integration functionalities.



If your application doesn't protect all the URIs you must ensure that such mappings are blocked to prevent unauthorized access.

Since version 3.4.0 the mappings are defined as follow:

- /ck/standard/filemanager
- /ck/standard/uploader
- /ck/ofm/filemanager
- /ck/ofm/filetree

in version 3.3.2:

- /ckconnector
- /ckuploader
- /ofm/filemanager
- /ofm/filetree

in versions before 3.3.2:

- /ckconnector
- /ckuploader

The controllers names are:

since version 3.4.0:

- openFileManagerConnector
- standardFileManagerConnector

in version 3.3.2:

- ckeditor
- openFileManagerConnector

in versions before 3.3.2:

- ckeditor

Also you can create filter that checks userSpace

```

class OfmSecurityFilters {
  def springSecurityService
  def filters = {
    ckOfm(controller: 'openFileManagerConnector', action: '*') {
      before = {
        if (springSecurityService.currentUser?.id != params.space?.toLong()) {
          response.sendError(403)
          return false
        }
      }
    }
    ckUpload(controller: 'standardFileManagerConnector', action: '*') {
      before = {
        if (springSecurityService.currentUser?.id != params.space?.toLong()) {
          response.sendError(403)
          return false
        }
      }
    }
  }
}

```

# Chapter 6. Advanced topics

## 6.1. Custom upload types

If you need to handle different types of uploads, since version 3.4.0, you can simply define a new block into the upload section:

```
...
upload {
  ...

  avatar {
    browser = true
    upload = true
    allowed = ['gif', 'jpg']
    denied = ['exe', 'sh', 'cgi']
  }

  ...
}

...
```

## 6.2. Internal URI's customization

Since version 3.4.0.1 it is possible to customize the URI's used to invoke the plugin's connectors.

Eg:

```
ckeditor.connectors.prefix = "/my/app/ck/"
```

# Chapter 7. History

- 3.6.6.1.1 (??/07/2013)
  - Fix for [GPCKEDITOR-37](#)
- 3.6.6.1.0 (15/07/2013)
  - Fix for [GPCKEDITOR-28](#)
  - Fix for [GPCKEDITOR-34](#)
  - Fix for [GPCKEDITOR-32](#)
  - Fix for [GPCKEDITOR-35](#)
- 3.6.4.0 (21/09/2012)
  - Fix for [GPCKEDITOR-26](#)
  - Fix for [GPCKEDITOR-27](#)
- 3.6.3.0 (28/04/2012)
  - Fix for [GPCKEDITOR-24](#)
- 3.6.2.2 (28/04/2012)
  - Fix for [GPCKEDITOR-18](#)
  - Fix for [GPCKEDITOR-29](#)
  - Fix for [GPCKEDITOR-20](#)
  - Fix for [GPCKEDITOR-21](#) (Thanks to Teppo Peltonen)
  - Fix for [GPCKEDITOR-22](#) (Thanks to Teppo Peltonen)
  - Fix for [GPCKEDITOR-23](#) (Thanks to Teppo Peltonen)
  - Fix for [GPCKEDITOR-25](#)



Breaking change: the standard (legacy) file manager has been removed ([GPCKEDITOR-25](#))

- 3.6.2.1 (24/01/2012)
  - Fix for [GPCKEDITOR-15](#)
- 3.6.2.0 (24/01/2012)
  - Upgrade to CKEditor 3.6.2
- 3.6.0.0 (04/06/2011)
  - Upgrade to CKEditor 3.6
  - Upgrade to latest Open File Manager
  - Added viewMode tag to start the file manager in list or grid mode

- Fix for [GPCKEDITOR-13](#)
- 3.5.4.1 (26/05/2011)
  - Fix for [GPCKEDITOR-13](#)
- 3.5.4.0 (21/05/2011)
  - Upgrade to CKEditor 3.5.4
- 3.5.3.0 (21/05/2011)
  - Upgrade to CKEditor 3.5.3
  - Upgrade to latest Open File Manager
- 3.5.2.0 (27/02/2011)
  - Upgrade to CKEditor 3.5.2
  - Upgrade to latest Open File Manager
  - Upgrade to Grails 1.3.7
  - Added id attribute to editor tag (Thanks to Paulo Alexandre Freitas)
  - Added removeInstance attribute to editor tag (Thanks to Paulo Alexandre Freitas)
- 3.4.1.0 (30/09/2010)
- Upgrade to CKEditor 3.4.1
- 3.4.0.2 (21/09/2010)
  - Fixed file manager behaviour in windows
- 3.4.0.1 (05/09/2010)
  - Added CKEditor sources
  - Added minified attribute to resources tag
  - Several fixes on Open File Manager
  - Added internal URI's customization
- 3.4.0 (25/08/2010)
  - Upgrade to CKEditor 3.4
  - Added defaultFileBrowser setting
  - Added support for custom upload types
  - Added support for callback js function on file selection in stand alone file browser
- 3.3.2 (12/08/2010)
  - Upgrade to Grails 1.3.4
  - Upgrade to CKEditor 3.3.2
  - Fix for [GRAILSPUGINS-1946](#) (Open File Manager integration)



- Fix for [GRAILSPLUGINS-1948](#)
- Fix for [GRAILSPLUGINS-2242](#)
- Fix for [GRAILSPLUGINS-2292](#)
- Fix for [GRAILSPLUGINS-2312](#)
- Fix for [GRAILSPLUGINS-2375](#)
- 0.6 (24/06/2010)
  - Upgrade to Grails 1.3.2
  - Upgrade to CKEditor 3.3.1
- 0.5 (02/06/2010)
  - Upgrade to Grails 1.3.1
  - Upgrade to CKEditor 3.3
- 0.4 (20/04/2010)
  - Upgrade to Grails 1.2.2
  - Upgrade to CKEditor 3.2.1
- 0.3 (10/02/2010)
  - Upgrade to Grails 1.2.1
  - Upgrade to CKEditor 3.1 ([GRAILSPLUGINS-1849](#))
  - Fix for [GRAILSPLUGINS-1942](#)
- 0.2 (05/01/2010)
  - Upgrade to Grails 1.2.0
  - Upgrade to CKEditor 3.0.2
- 0.1 (04/11/2009)
  - First public release