

# Plugins

## Table of contents

Plugins. . . . .	1
Plugin-template . . . . .	1

## Plugins

You can replace components with a plugin or add new features to Hawron. Have a look at the included plugins like Jetty or the viewer to see what is possible. Note, every plugin becomes an own classloader (searches child-first), that should avoid problems with different versions of libraries.

Here is a short list of interfaces which can be used.

- implement [de.miethxml.toolkit.plugins.Plugin](#) if you need the location of your plugin at runtime (for reading and storing files/resources)
- implement [de.miethxml.toolkit.components.PluginComponent](#) if you want override an existing component
- implement [de.miethxml.hawron.gui.context.editor.Editor](#) to provide a special content-editor
- implement [de.miethxml.hawron.gui.context.viewer.Viewer](#) to provide a special view
- implement [de.miethxml.hawron.gui.context.action.Action](#) to provide a special Action

Look at [Avalon](#) to find out more about the lifecycle of components.

The following descriptor-example shows which settings are needed.

- The "instanceclass" is your implementation of one of the supported interfaces.
- The "interface" is needed for the PluginReceiver, which listen for this plugin (as example the EditorManager).
- Add all needed libraries as a classpath-element, relative to your plugin-directory

```
<?xml version="1.0" ?>
<plugin>
  <name>My plugin</name>
  <instanceclass>MyPluginImpl</instanceclass>
  <interface>de.miethxml.hawron.context.viewer.Viewer</interface>
  <description>My plugin</description>
  <resources>
    <classpath src="my.jar"/>
  </resources>
</plugin>
```

## Plugin-template

```
ant newplugin -Dplugin.name=foo
```

Will generate a new subdirectory "plugins/other/foo" and place there a generic buildfile and a plugins-descriptor. Optional you can add

-Dplugin.category=bar

this will create a subdirectory "plugin/bar/foo". Copy the plugin.properties to local.plugin.properties and uncomment your plugin there for building.