Joomla! Override Plugin

What is an override?

There may be occasions where you would like to change the way a Joomla! Extension (such as a Component or Module, whether from the Joomla! core or produced by a third party) works; or you want to create new functionalities in your site, but you don't want to lose those changes when the component is updated nor do you want to stop updating Joomla.

The standard code from any Joomla! Module or Component can be overridden by adding/ extending code to Joomla's “override” directory or your template's "code" directory, depending on your override scope (see below). This is a very simple, yet powerful, way to create overrides.

What is an override scope?

You have two scope types for extending and overriding extensions:
1) A template override scope
2) A global override scope

The template override scope only applies when the respective template is active. While the global override scope applies to all templates. This approach gives the site developer a lot of flexibility and power. You can also use both template override override scope and global override scope to create special functionalities for each template.

How does the plugin work?

Template override scope - It works similar to the standard functionality to override layout outputs. You can override extension code under a template folder, by adding a “code” folder.

For example:
root/templates/your_template/code/com_content/controller.php
root/templates/your_template/code/administrator/com_content/controller.php

If you are working with two or more templates and need to override an extension for all your templates, the best option is use the global scope.

Global override scope - The global override scope is as easy as templating, but is done in the “override” folder within the Joomla root. If you want to override fronted extensions, create an “override” folder in joomla root directory. If you want to override an administrator extension, create an “administrator” folder within the “override” folder.

For example:
root/override/com_content/controller.php
root/override/administrator/com_content/controller.php

What is the syntax for the Template Override Scope?

Override base component controller:
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/CONTROLLER.php

Override another component controller:
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/controllers/
CONTROLLER_NAME.php

Override models:
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/models/MODEL_NAME.php

Override forms:
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/models/forms/
FORM_NAME.php

Override views:
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/views/VIEW_NAME/
VIEW_NAME(.FORMAT).php

Override views templates:
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/views/VIEW_NAME/tmpl/
TEMPLATE_NAME.php

Override Module Manifest
root/templates/TEMPLATE_NAME/code/MOD_NAME/MOD_NAME.xml

Override Logic
root/templates/TEMPLATE_NAME/code/MOD_NAME/MOD_NAME.php

Override Template
root/templates/TEMPLATE_NAME/code/MOD_NAME/tmpl/TEMPLATE_NAME.php

What is the syntax for the Global Override Scope?

Site general override:
root/override/COMPONENT_NAME/CONTROLLER.php

Override another component controller:
root/override/COMPONENT_NAME/controllers/CONTROLLER_NAME.php

Override models:
override/COMPONENT_NAME/models/MODEL_NAME.php

Override forms:
root/override/COMPONENT_NAME/models/forms/FORM_NAME.php

Override views:
root/override/COMPONENT_NAME/views/VIEW_NAME/VIEW_NAME(.FORMAT).php

Override view template:
root/override/COMPONENT_NAME/views/VIEW_NAME/tmpl/TEMPLATE_NAME.php

Override Module Manifest
root/override/MOD_NAME/MOD_NAME.xml

Override Logic
root/override/MOD_NAME/MOD_NAME.php

Override Template
root/override/MOD_NAME/tmpl/TEMPLATE_NAME.php

Administrator general override:
root/override/administrator/COMPONENT_NAME/CONTROLLER.php

Override another component controller:
root/override/administrator/COMPONENT_NAME/controllers/CONTROLLER_NAME.php

Override models:
root/override/administrator/COMPONENT_NAME/models/MODEL_NAME.php

Override forms:
root/override/administrator/COMPONENT_NAME/models/forms/FORM_NAME.php

Override views:
root/override/administrator/COMPONENT_NAME/views/VIEW_NAME/VIEW_NAME(.FORMAT).php

Override view template:
root/override/administrator/COMPONENT_NAME/views/VIEW_NAME/tmpl/TEMPLATE_NAME.php

Override Module Manifest
root/override/administrator/MOD_NAME/MOD_NAME.xml

Override Logic
root/override/administrator/MOD_NAME/MOD_NAME.php

Override Template
root/override/administrator/MOD_NAME/tmpl/TEMPLATE_NAME.php

What is the execution priority?

The execution priority is as follows:
1) Template override scope
2) Global override scope
3) Joomla core

For example, if we have the following controller.php files:

```
root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/CONTROLLER.php
root/override/COMPONENT_NAME/CONTROLLER.php
root/components/COMPONENT_NAME/CONTROLLER.php
```

the one within the template takes priority and it overrides the other two controller.php files. This example assumes the respective template is loaded for the page. If another template is loaded for the page that does not have an override for that file, the global override scope is used instead.

If there are no overrides, the Joomla core is used.

How do you extend default component code?

When you're working with overriding component code you use OOP. You could replace the original class instead of extending it, but most in most cases it's preferable to extend it. This way you only have to override the methods that you want to customize.

Extending a Component Controller Class example:

```
class ComponentControllerName extends ComponentControllerNameDefault
{

}
```

Extending a Component Model Class example:

```
class ComponentModelName extends ComponentModelNameDefault
{

}
```

Extending a Component View Class example:

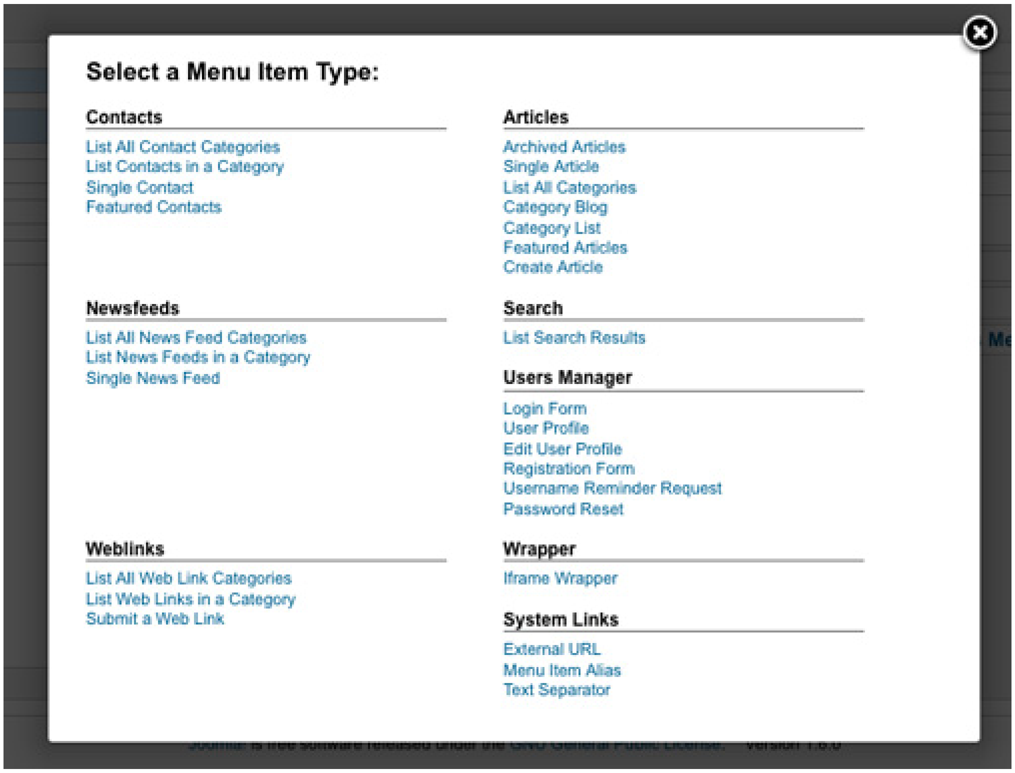
```
class ComponentViewName extends ComponentViewNameDefault
{

}
```

How do you add a new view to a component?

Sometimes you need to create new functionalities for a component.

How do you add a new view for a frontend menu item?

In the Joomla framework, components are executed using menu items. If you go in the menu manager of your Joomla installation, you will see "menu types" grouped by component.



Select a Menu Item Type:

Contacts

List All Contact Categories
List Contacts in a Category
Single Contact
Featured Contacts

Articles

Archived Articles
Single Article
List All Categories
Category Blog
Category List
Featured Articles
Create Article

Newsfeeds

List All News Feed Categories
List News Feeds in a Category
Single News Feed

Search

List Search Results

Users Manager

Login Form
User Profile
Edit User Profile
Registration Form
Username Reminder Request
Password Reset

Weblinks

List All Web Link Categories
List Web Links in a Category
Submit a Web Link

Wrapper

Iframe Wrapper

System Links

External URL
Menu Item Alias
Text Separator

If you want to create new menu item type which you can select for menu items,, simply create a file based on your scope.

Template Override Scope

root/templates/TEMPLATE_NAME/code/COMPONENT_NAME/views/VIEW_NAME/tmpl/TEMPLATE_NAME.php

Global Override Scope

root/override/COMPONENT_NAME/views/VIEW_NAME/tmpl/TEMPLATE_NAME.xml

containing:

```
<?xml version="1.0" encoding="utf-8"?>
<metadata>
	<layout
title="COM_COMPONENT_NAME_VIEW_VIEWNAME_TEMPLATENAME_TITLE">
		<message>
			<!
[CDATA[COM_COMPONENT_NAME_VIEW_VIEWNAME_TEMPLATENAME_DESC]]>
		</message>
	</layout>
</metadata>
```

remember this is an basic example, you can add parameters also.

How do you create custom code that runs when accessing a component in the administrator?

This code can be useful for

- adding submenu in administrator component
- adding custom css/javascript
- run your custom sql file to create your new tables