Joomla! Development using PhpStorm

Redirection Notice
This page will redirect to https://www.jetbrains.com/help/phpstorm/joomla-specific-coding-assistance.html in about 2 seconds.

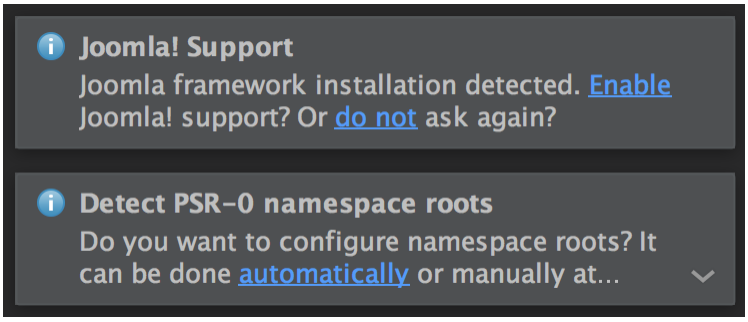
Tweet

This tutorial describes the features and best practices in using PhpStorm as an IDE for Joomla development. Starting with version 2016.1.1, PhpStorm bundles the Joomla! Plugin providing many Joomla!-specific features.

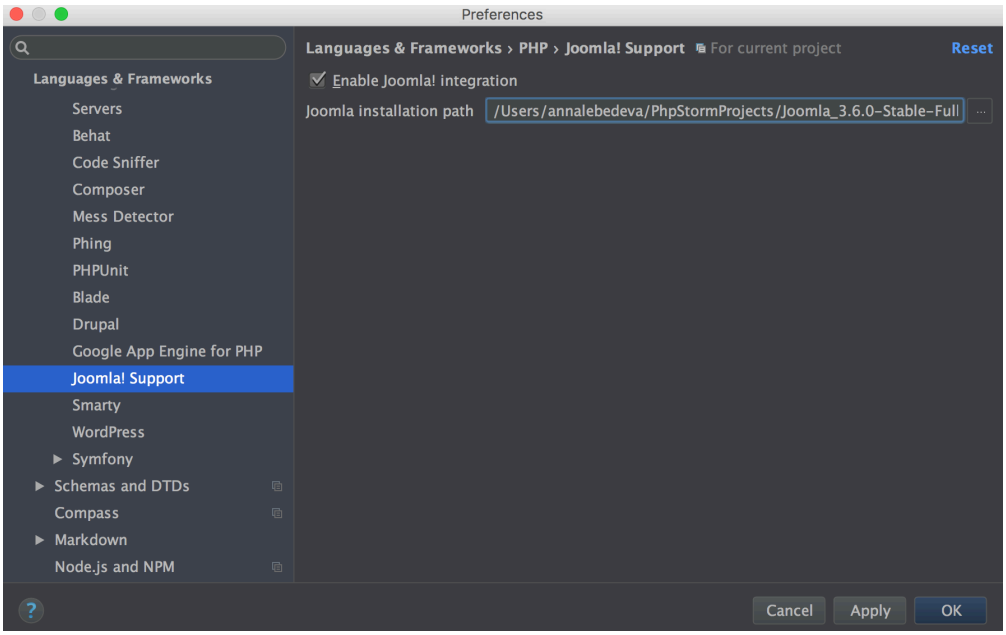
Joomla is an award-winning content management system (CMS), which enables you to build websites and powerful online applications. Many aspects, including its ease-of-use and extensibility, have made Joomla the most popular website software available. Best of all, Joomla is an open source solution that is freely available to everyone.

Enabling Joomla! Integration in an Existing PhpStorm Project

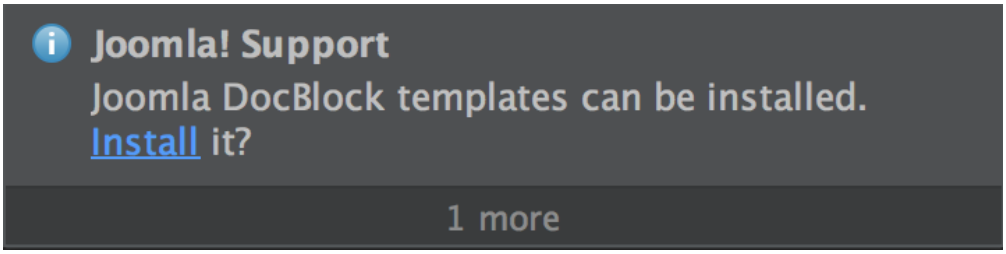
When we open an existing Joomla! Project in PhpStorm, PhpStorm should detect that we're working on a Joomla! application and ask us to enable the Joomla! integrations.



When we click Enable, PhpStorm will ask us to browse to the location where the Joomla! codebase is installed. In my case, it is the root of the project I'm working on. It's also worth clicking the second prompt that asks you if you want to detect the namespace roots; this saves us having to configure the paths for the project manually. If you don't get prompted by the Joomla! Support prompt, you can enable Joomla! support manually by opening the preference pane, and navigating to Languag es & Frameworks then PHP and Joomla! Support.

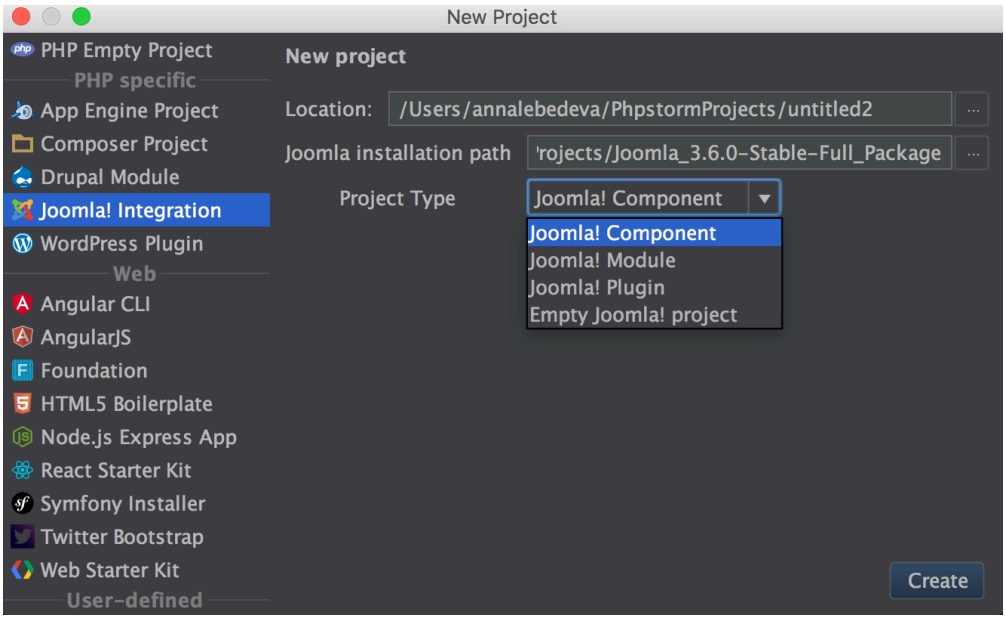


Once we've enabled Joomla! support, PhpStorm asks if we would like to enable the Joomla! code styles and if we would like to enable the Joomla! docblock templates. Joomla! has its own rigorous code style and docblocks, so enabling these is always a good idea; we'll learn more about these later.



Creating a New Joomla Module/Plugin/Component/Empty Joomla! Project

To create a new Joomla project, go to File then New Project... and select Joomla! Integration from the left-hand options. You'll need to tell PhpStorm where to find your Joomla! installation path (unless you've already specified it at the previous step), give your project a name, and select what type of Joomla! project you're going to create. Here we're creating a new Joomla! module.

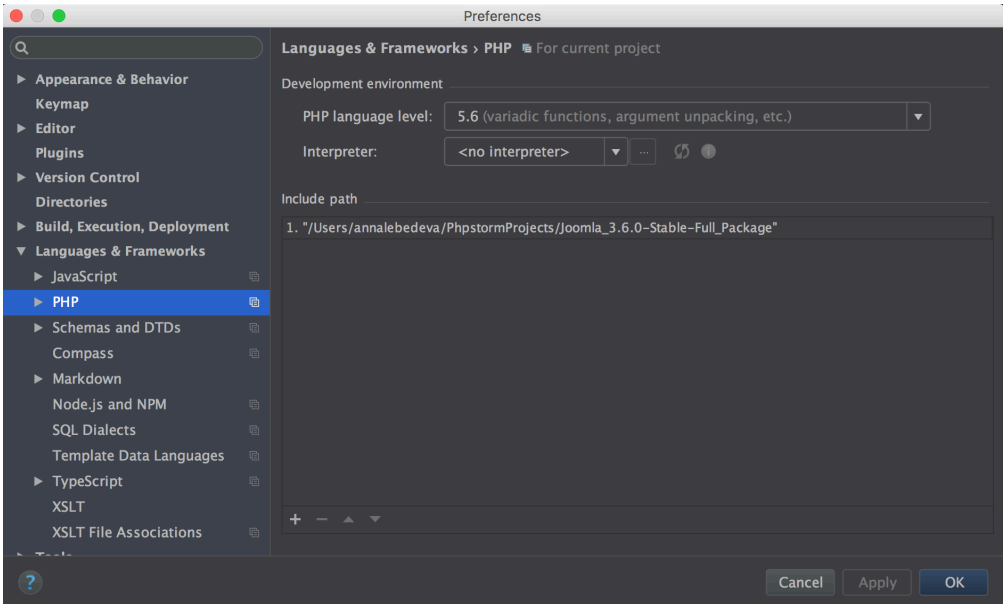


PhpStorm will then create the `joomla-module.php` file, and the `joomla-module.xml` file that you need to use this project as a Joomla! module. You'll need to edit the configuration XML file before you can use the module.

Development Environment Configuration

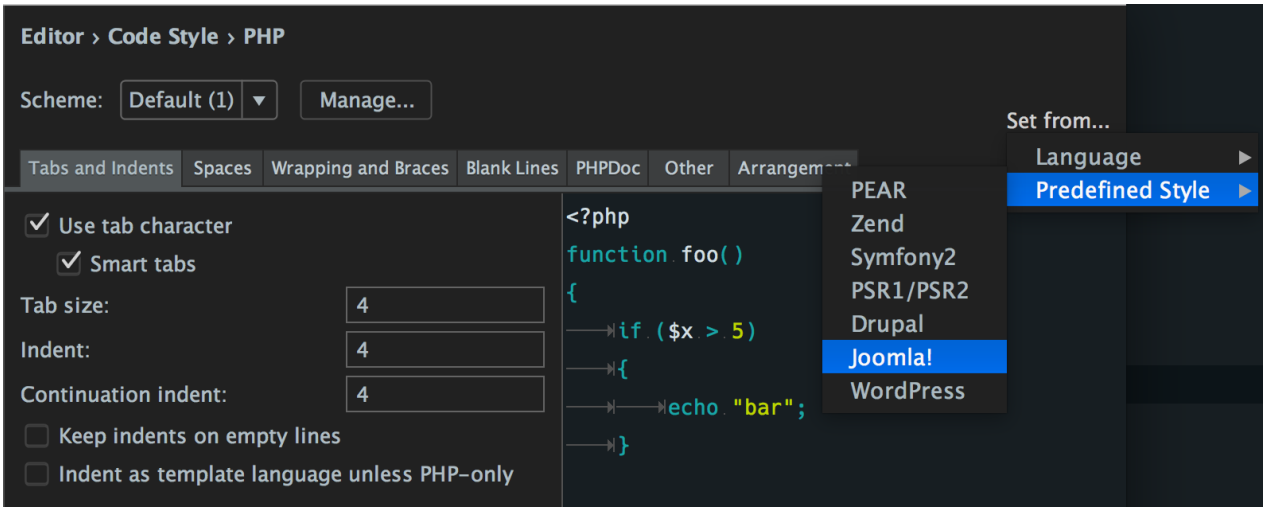
Joomla! Include Paths

With Joomla! support enabled and PhpStorm knowing where your Joomla! install is, the path to this Joomla! install should be included in your include paths by default. You can check this by selecting External Libraries in the left-hand project browser, and check that you can see joomla library root under PHP. If you don't see this, you'll need to add the path to your Joomla! install to your project manually in Settings | PHP | Include path.



Joomla! Coding Standard (Code Style)

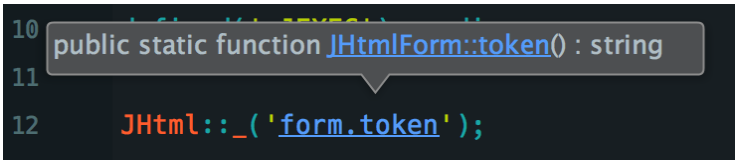
PhpStorm ships with a coding standard for Joomla! code style, and you should be asked if you want to enable this when you enable Joomla! support. If you wish to enable the Joomla! code style manually, browse to Editor, Code Style then PHP in the preference pane. There, you can add the pre-defined Joomla! styles by clicking Set From..., and selecting Predefined Style then Joomla!



JHtml::_($argument) and JText::_($argument) support

JHtml::_($argument) support

The latest versions of PhpStorm come with support for the ` _` static method of the `JHtml' class. This is a magic method that allows you to invoke helpers by passing in a string that contains the class name plus the method name, separated by a dot. PhpStorm can parse these strings, and provide you with the usual helpers associated with class and method names in the IDE. For example, here we are using the `form.token` helper, and when we invoke Brief Info over the string (default to CMD/CTRL plus hover over the item), we see the method signature for this helper method:



Holding CMD/CTRL and clicking the string takes us straight to the `token` static method of the `JHtmlForm` class just as we would expect. Of course, the `JHtml` class comes with full code completion and type hinting like all classes do in PhpStorm.

JText::_($argument) support

Similarly to JHtml support, PhpStorm now also supports the static methods of the `JText` class. Text allows you to handle translations from definitions (typically in .ini files) and output the translation in the correct place. When you use the `_` or `sprintf` or 'script' methods of `JText` and pass in a valid string with a key name, PhpStorm will allow you to use Brief Info an d navigate to the definition (using CMD/CTRL and click) to be taken directly to the .ini file that defined that key.

Databases

Data source detection

PhpStorm comes with a database browser built right in, and the Joomla! integrations allow us to quickly and easily configure the database tool from the configuration file that Joomla! creates containing our credentials. Once we've opened the Database tool (I usually hover over the menu icon in the bottom left of the screen and select Database), we can add a new data source for our Joomla! database by clicking the + icon and then selecting Import from sources...

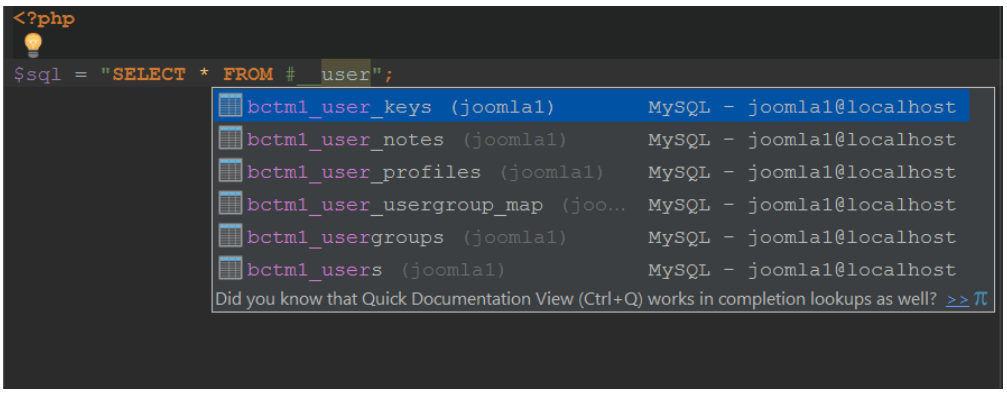
We then see the Add New Datasource dialog, but with all the fields completed from the settings in the `configuration.php` file.

Simply click Test Connections to check everything is working, and then click OK.

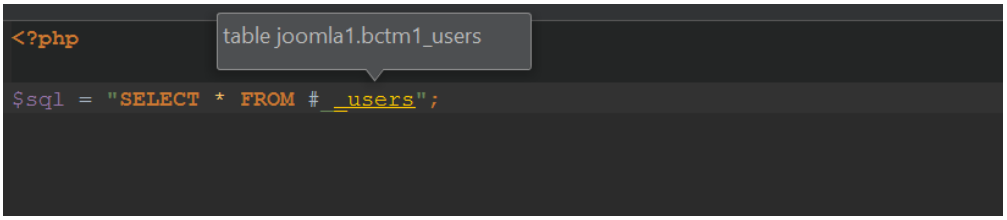
Database prefix support

You can see that Joomla! adds a prefix to the database tables (which is generated or configured during the installation wizard), and this can make writing queries in the query editor (part of the database tool) quite painful.

PhpStorm provides database prefixes support and changes #__ on fly to the prefix that is defined in the $dbprefix field in the c onfiguration.php file. You will also have completion working for your SQL queries.To have it working, make sure that SQL Dialect for project is equal to Database (Settings-> SQL Dialects).



If you Ctrl+Click, you can navigate directly to the table from the query.

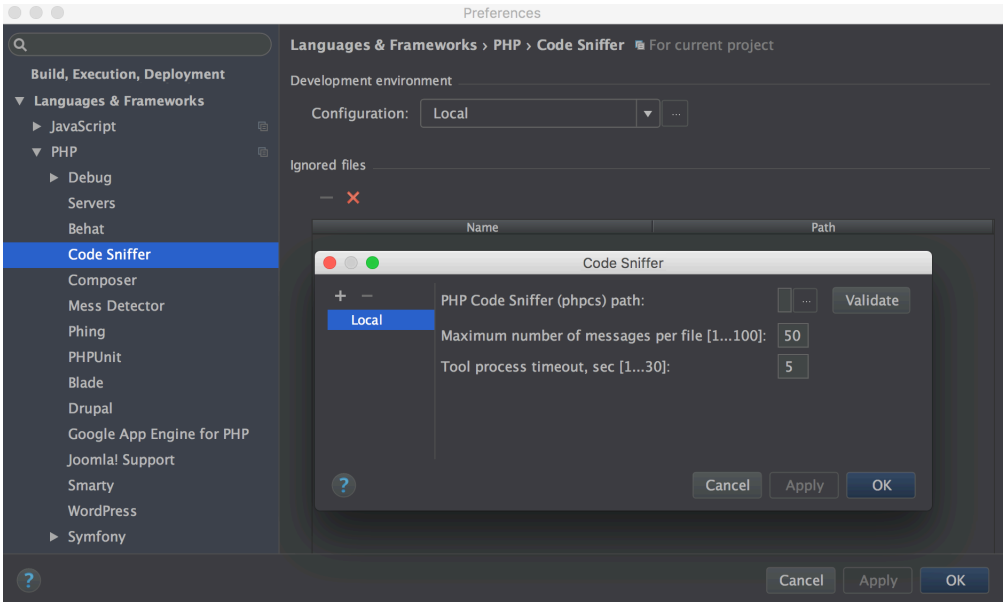


Joomla! Doc-blocks standard support

Joomla! code styles have strict standards about docblocks - including which docblock annotations are required, which are optional, and which order they should be presented in. PhpStorm now ships with an inspection that will tell you exactly what is wrong with your docblocks, and why they don't meet Joomla! strict standards.

Joomla! CodeSniffer

Joomla! provides a ruleset for use with PHP CodeSniffer, and PhpStorm comes with support for PHP CodeSniffer out of the box.



Refer to the PHP Code Sniffer in PhpStorm tutorial with the Joomla! CodeSniffer standards to add more inspections for code style within PhpStorm.

Debugging and Profiling Joomla! Applications with PhpStorm

Joomla!-based projects can be debugged and profiled without any Joomla!-specific configuration.

Please proceed with standard PhpStorm debugging or profiling workflow. For additional details on debugger and profiler configuration, check this tutorial or relevant videos on the video tutorials page .

Download latest version of PhpStorm for your platform right now >>

Tweet