



Institut für  
Informationswissenschaften

## Institute for Information Science Development Guide

James Antrim, Mariusz Homeniuk, Mehmet Ali Pamukci,  
Dennis Priefer, Wolf Rost, Niklas Simonis, Dennis Voigtländer

August 25, 2017

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>The Development Environment</b>	<b>5</b>
2.1	Local Web Server Installation and Configuration . . . . .	5
2.1.1	XAMPP . . . . .	5
2.1.2	PHP Configuration . . . . .	6
2.1.3	PHPUnit . . . . .	8
2.2	Git . . . . .	9
2.2.1	Installation . . . . .	9
2.2.2	User Configuration . . . . .	9
2.3	PhpStorm . . . . .	10
2.3.1	Installation . . . . .	11
2.3.2	Required Plugins . . . . .	12
2.3.3	Gerrit Configuration . . . . .	14
2.3.4	THM Core Library . . . . .	15
2.3.5	Tool Window Activation . . . . .	16
2.3.6	Deployment - Mapping . . . . .	16
2.4	Code Sniffer and Mess Detector . . . . .	20
2.4.1	Standards Directory Inclusion . . . . .	21
2.4.2	PhpStorm . . . . .	21
2.5	SASS . . . . .	23
2.5.1	Installation . . . . .	24
2.5.2	PhpStorm Configuration . . . . .	24
2.6	Joomla! . . . . .	28
2.6.1	Database Creation . . . . .	28
2.6.2	MNI Site Development . . . . .	29
2.6.3	Joomla! 3.x Development . . . . .	31
<b>3</b>	<b>Tool Guide</b>	<b>34</b>
3.1	Git . . . . .	34
3.2	Gerrit . . . . .	34
3.2.1	Workflow . . . . .	35
3.2.2	Cloning repository . . . . .	35
3.2.3	Push to repository . . . . .	35
3.2.4	Review change . . . . .	37

3.2.5	Submit/Abandon change	38
3.2.6	FAQ and Help	39
<b>4</b>	<b>Development Guide</b>	<b>40</b>
4.1	Workflow	40
4.2	Developing	41
4.2.1	Advanced PHP Tips	41
4.3	Integration	41
4.3.1	Reviewing	41
4.4	Testing	44
4.5	Debugging	45
<b>A</b>	<b>How-tos</b>	<b>48</b>
A.1	How to change environment variables in Windows	48

# Chapter 1

## Introduction

This documents explains how to set up and develop in the context of the iCampus Working Group. All instructions in this document are based upon the Windows 8.1 operating system.

## Chapter 2

# The Development Environment

In order to develop on a personal computer, must certain steps be performed in various systems, as well as various programs installed and configured. In this chapter these items will be discussed.

### 2.1 Local Web Server Installation and Configuration

In order for a locally developed project to be deployed, and to guarantee access to all required components a web server must be installed locally.

#### 2.1.1 XAMPP

In the iCampus web development group we use XAMPP from Apache Friends for our local web servers. Apache Friends describes XAMPP as follows:

*XAMPP is a completely free, easy to install Apache distribution containing MySQL, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.*<sup>1</sup>

As per the description the distributions of XAMPP available from [Apache Friends](https://www.apachefriends.org/index.html) come included with various useful tools which are also used within iCampus, such as PHP (server-side scripting language) and MySQL (database management system), as well as many other tools which are optional for iCampus such as FileZilla (FTP Server) and Mercury (Mail Server).

Download and install the latest XAMPP installation. Keeping in mind that at least MySQL and phpMyAdmin need to be installed to function in the iCampus environment.

---

<sup>1</sup><https://www.apachefriends.org/index.html> September 19, 2014

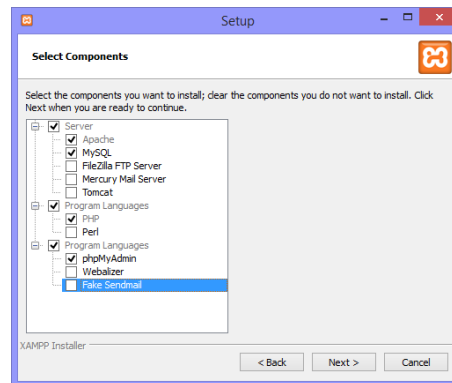


Figure 2.1: XAMPP Minimum Installation

### 2.1.2 PHP Configuration

The next step is the configuration of PHP for error reporting using xdebug. To do this open your `php.ini` file using the XAMPP control panel or the file explorer. This file is typically located at <C:/xampp/php/php.ini>.

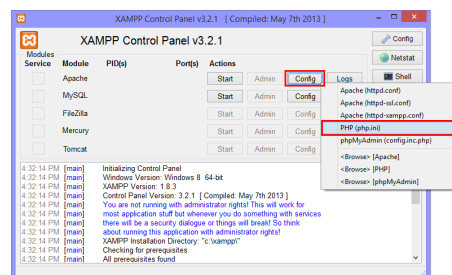


Figure 2.2: XAMPP Control Panel

The following modifications should then be made to the file:

```
// Reports all PHP Errors, Warnings, and Breaches of PHP Standards
error_reporting = E_ALL | E_STRICT
```

```
// Turns off error buffering => errors reported immediately, necessary for the debugger
output_buffering = Off
```

```
// Debugger Settings (may only need to be commented in)
[XDebug]
zend_extension="C:\xampp\php\ext\php_xdebug.dll"
xdebug.remote_enable=true
xdebug.remote_host=localhost
xdebug.remote_port=9000
xdebug.remote_handler=dbgp
xdebug.profiler_enable=1
xdebug.profiler_output_dir="C:\xampp\tmp"
```

If you have trouble to get XDebug to work, this page (<https://xdebug.org/wizard.php>) might help you. You have to copy your whole phpinfo()-page (<http://localhost/dashboard/phpinfo.php>) and paste it into the textarea on the xdebug-wizard page. Follow the instructions and you should be ready to use xdebug.

The interface shown in Figure 2.2 above will also later used to start and stop the server service itself, Apache, as well as the the supporting database management service, MySQL, by pressing the corresponding “Start”/“Stop” button.

### 2.1.3 PHPUnit

The following text describes the installation of PHPUnit.<sup>2</sup>

#### For Linux Users

```
wget https://phar.phpunit.de/phpunit.phar
chmod +x phpunit.phar
sudo mv phpunit.phar /usr/local/bin/phpunit
```

#### For Windows Users

1. Create a directory for PHP binaries; e.g., C:\bin.
2. Add your PHP directory C:\xampp\php and the above created directory to your system's PATH environment variable. See appendix A.1.
3. Download <https://phar.phpunit.de/phpunit.phar> and save the file as C:\bin\phpunit.phar
4. Open a command line (e.g., press Windows+R > type cmd > ENTER)
5. Create a wrapping batch script (results in C:\bin\phpunit.cmd):

```
C:\Users\username> cd C:\bin
C:\bin> echo @php "%~dp0phpunit.phar" %* > phpunit.cmd
```

Open a new command line and confirm that you can execute PHPUnit from any path and it is the right PHPUnit version you installed previous:

```
> phpunit --version
> PHPUnit x.y.z by Sebastian Bergmann.
```

---

<sup>2</sup>For the original documentation see <https://phpunit.de/manual/current/en/installation.html>



## 2.2 Git

For repository and versioning iCampus uses Git. To this end Git must first be downloaded from [Git](#), installed and configured.

### 2.2.1 Installation

#### Command Context

Part of the configuration is deciding which context you would like to be able to use Git in. In iCampus the second is used because it allows the developer later to use PHPStorm's own terminal to execute Git's commands, which eliminates the necessity for navigation to projects in Git Bash or Windows shell. More on PHPStorm in Section [2.3](#).

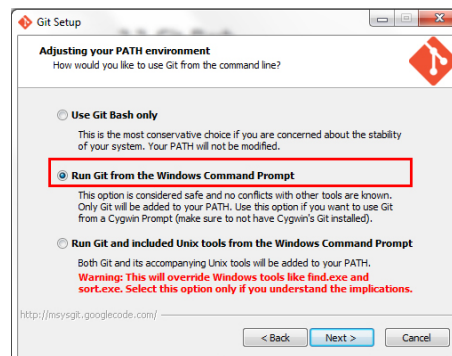


Figure 2.3: Command Context

#### Line Endings

The next screen lets one decide how file endings are pulled and committed. The iCampus Coding standards demand the Unix-style line endings. To this end only the second option is sufficient, because once style checking is enabled in your IDE the first and third options will, dependent on your system settings, report every single line of code as a breach of style, which removes any transparency whatsoever when searching for other errors.

### 2.2.2 User Configuration

In order to take part in the development process you must take several steps to configure your user account.

#### Global User Configuration

In Git Bash the global user name and email must be set so that repository actions can be associated with a user.

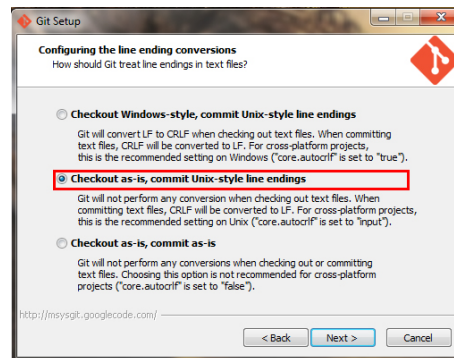


Figure 2.4: Line Endings

Start Git Bash and enter the following commands, replacing <First Name>, <Last Name>, and <Department> with your first and last names and the abbreviation of the department in which you are matriculated.

```
git config --global user.name "<First Name> <Last Name>"
git config --global user.email "<First Name>.<Last Name>@<Department>.thm.de"
```

You can confirm that the entries were saved correctly by entering the following command and comparing the associated values.

```
git config -l
```

## SSH-Key Creation

After the user configuration is completed, an SSH-Key can be created using the following command in Git Bash.

```
ssh-keygen
```

The first question determines where the generated key will be saved and what its name will be. Per default this will be `C:/Users/<Windows Account Name>/.ssh/id_rsa`. This directory is important for the next steps, as well as later when repositories are cloned. Press enter to confirm the default. The next two questions will be to enter and confirm your password. You can also elect to have no password by pressing enter in answer to both questions.

## 2.3 PhpStorm

PhpStorm is the IDE used in iCampus. While others are allowed the use of a single IDE in the working group allows for the exchange of information and tips on how to get the most out of a single IDE. This makes everyone more productive and allows for a collaborative effort to solve any problems which may be IDE related.

PhpStorm was chosen for its ease of use and integration of many tools which simplify development, including CodeSniffer, database connection tools, SASS support, Git support, and many others. PhpStorm can be downloaded from the [JetBrains Website](#).

### 2.3.1 Installation

During the installation process you will be asked which file extensions should be associated with PhpStorm. As we use PhpStorm for the development with all given extensions, all can be selected. As of PhpStorm 8 this also allows for the editing of single files without the creation of a project.

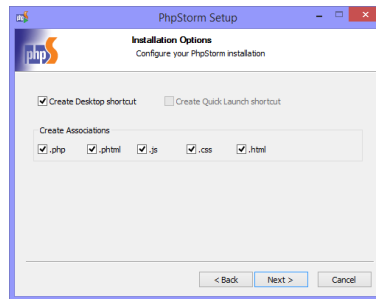


Figure 2.5: File Associations

For the most part the rest of the installation requires nothing special, however at the end you will be asked to choose between several themes and coloring/font style options. As this interface offers no preview it is best to go with the default settings. These selections can later be changed while running the PhpStorm which will actually show you what effects your selection has on its appearance.

After the installation starting PhpStorm will request the license information. Please register at <https://www.jetbrains.com/shop/eform/students> as a student. After this procedure you can activate PhpStorm using your JetBrains account.

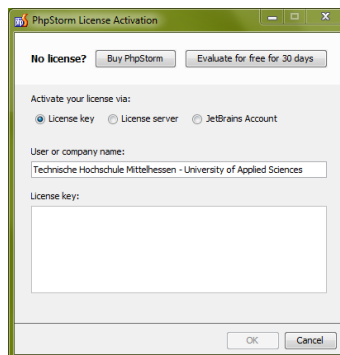
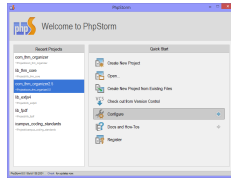


Figure 2.6: PhpStorm License Activation

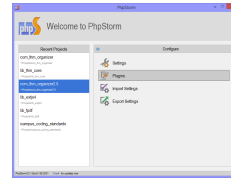
### 2.3.2 Required Plugins

Next we will need to install two plugins, Gerrit and Jenkins. Gerrit is a repository and code review system based on Git.<sup>3</sup> where Jenkins is a Continuous Integration (CI) tool which builds and tests projects.<sup>4</sup>

To do so we must first navigate to the plugins view. Open the PhpStorm plugin store by clicking **Configure** and then **Plugins** on the next screen. This can later be accessed through the menu under **File**  $\Rightarrow$  **Settings**  $\Rightarrow$  **Plugins** (under **IDE Settings**). The installation steps are also depicted below in Figures 2.7 and 2.8.

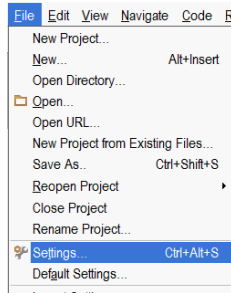


(a) Configure

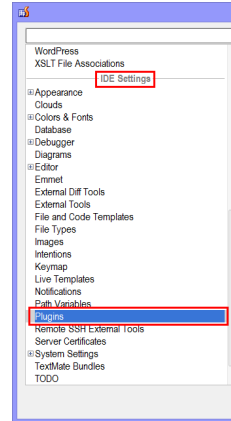


(b) Plugins

Figure 2.7: Welcome Screen Plugin Installation



(a) Settings



(b) Plugins

Figure 2.8: Project Plugin Installation

<sup>3</sup>For further information visit the Gerrit documentation page at [https://gerrit-review.googlesource.com/Documentation/intro-quick.html#\\_what\\_is\\_gerrit](https://gerrit-review.googlesource.com/Documentation/intro-quick.html#_what_is_gerrit)

<sup>4</sup>For further information visit the Jenkins documentation page at <https://wiki.jenkins-ci.org/display/JENKINS/Meet+Jenkins>

Then we need to access the plugin store to find yet uninstalled plugins To do so click **Browse repositories....**

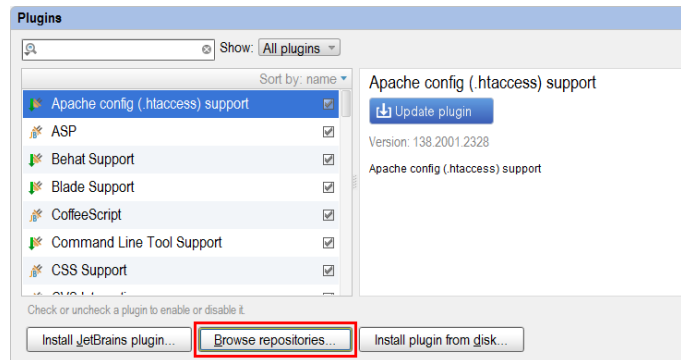


Figure 2.9: Browse Repositories

Once the plugin store is open enter “Gerrit” in the search to find the appropriate plugin. Then click on the green **Install plugin** button, or use the item’s context menu and click **Download and Install**.

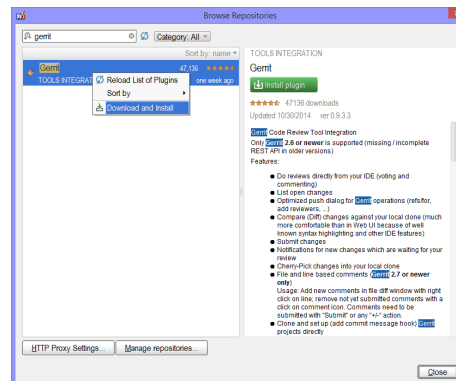


Figure 2.10: Gerrit Installation

You will then be prompted to restart PhpStorm to enable the plugin. **Do not restart yet.** Instead search the store for the Jenkins plugin in the same way as we just searched for Gerrit and install it. Now perform the restart.

### 2.3.3 Gerrit Configuration

Visit our Gerrit repository located at <http://gerrit.mni.thm.de/gerrit>. In the upper right corner is a link to **Sign In** to the server. Use your THM user credentials to log in.

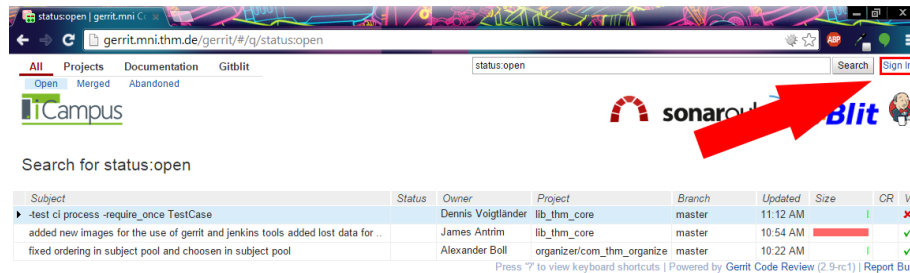


Figure 2.11: Gerrit Sign In

After signing in you need to create a HTTP password for yourself. Your name should be shown in the same position where **Sign In** stood before. Next to it is a small black arrow pointing down. Clicking this arrow opens a small window. Click **Settings**. This opens your account settings. Click **HTTP Password** then **Generate Password** in the view which opens. This generates the password in the field above the button. Copy this manually or use the button to the right of the field. This password can be viewed at anytime by navigating to the **HTTP Password** settings item.

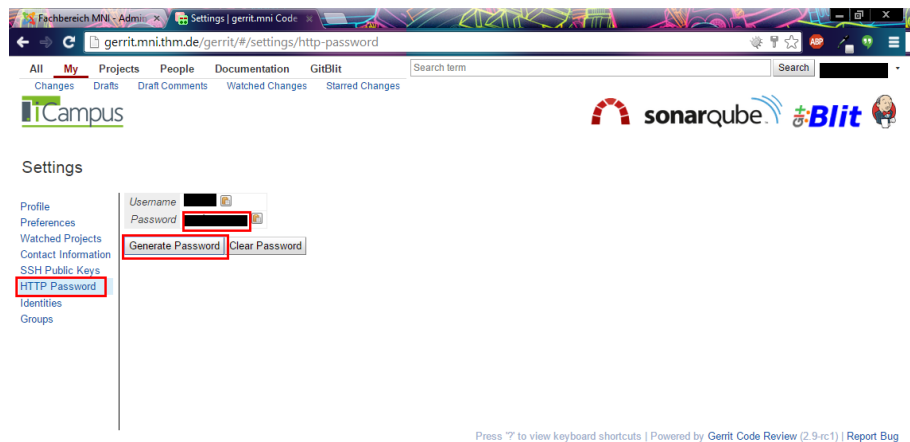


Figure 2.12: Gerrit HTTP Password Generation

Next we will checkout a project using the Gerrit plugin. Click **Checkout** from **Version Control** and then select Gerrit from the window which opens.

After making the selection the system will ask you to set a master password. **Leave these fields blank**. Setting a password will cause you to have to enter a it every time you perform writing repository actions. (This can later be changed.) After this you will then be prompted to enter

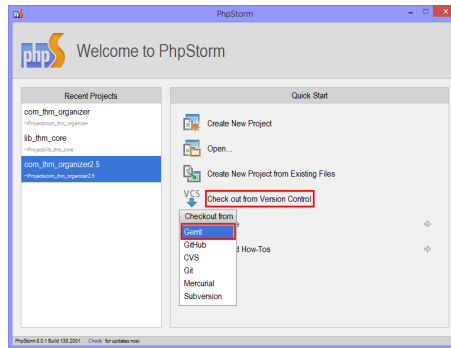


Figure 2.13: Check out a Gerrit Project

the connection information for Gerrit. This includes the “URL” (<http://gerrit.mni.thm.de/gerrit>), “Login” (your THM username), and “Password” (the HTTP Password generated).

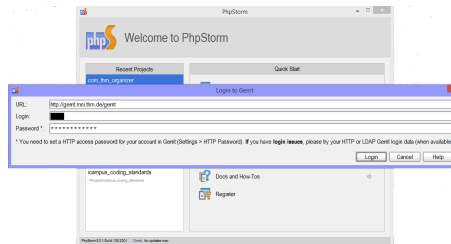


Figure 2.14: Gerrit Connection Setup

Should everything be correctly entered we can now checkout a project.

### 2.3.4 THM Core Library

The project we will be checking out is the THM Core Library (`lib.thm.core`). The THM Core Library carries with it many useful functions helpful in developing Joomla! extensions as well as documentation (including this document), styles, coding standards and much more.

For the “Git Repository Url” select <http://gerrit.mni.thm.de/gerrit/lib.thm.core>. After this PhpStorm may complain that the PhpStormProjects directory does not exist. Either create the default “PhpStormProjects” at the location shown or chose a different one into which this (and your future projects if you so chose) will be saved.

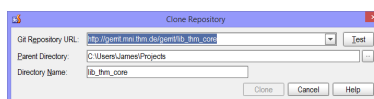


Figure 2.15: Clone Repository

### 2.3.5 Tool Window Activation

For quick access to the Gerrit and Jenkins plugins it is helpful to add their tool windows to the PhpStorm displays. To do so click on **View** → **Tool Windows** and then click on **Gerrit** and **Jenkins** in turn.

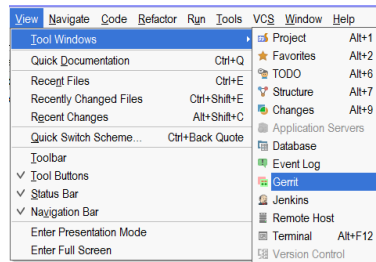


Figure 2.16: Tool Window Activation

This will add icons with text to the borders of the PhpStorm window. Clicking these opens the corresponding tool windows.

### 2.3.6 Deployment - Mapping

This section will describe how to configure deployments in PHPStorm. So the files and folder will be automatically synced between your project folder and Joomla instance.

- **Step 1:** External Libraries — > right-click — > Configure Php Include Paths  
Alternatively : F4 key

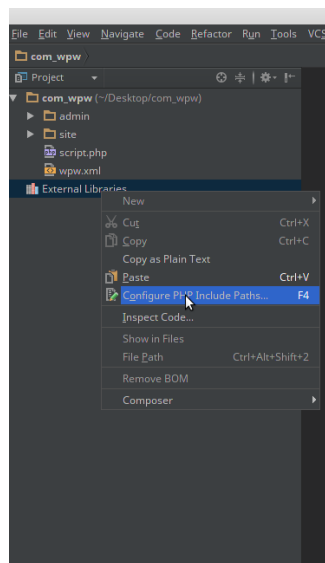


Figure 2.17: Configure external libraries



- **Step 2:** Click on the green plus icon.

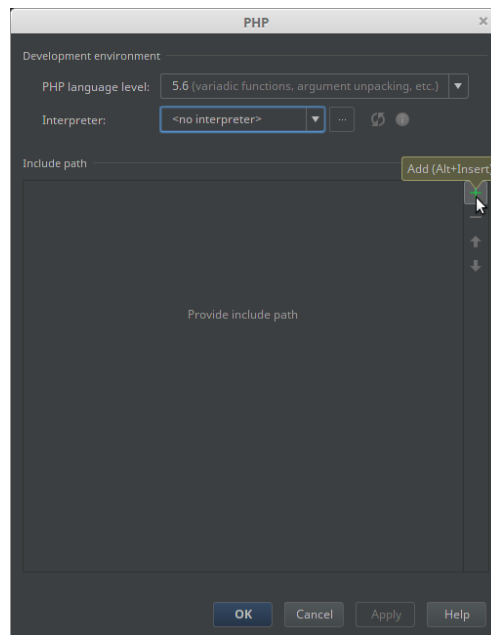


Figure 2.18: Add a Joomla instance

- **Step 3:** Choose the path to your Joomla instance.

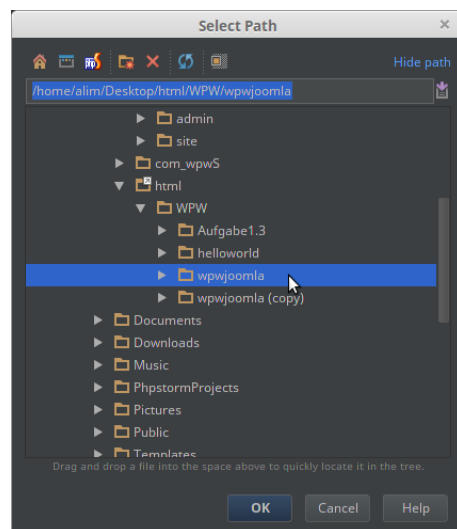


Figure 2.19: Choose the path to your local Joomla instance

- Step 4: File –> Settings

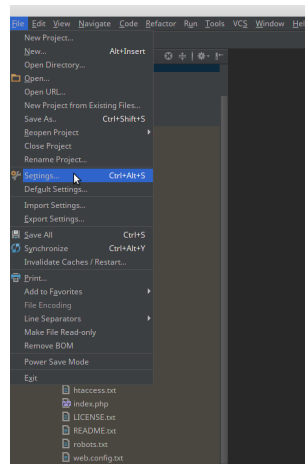


Figure 2.20: Deployment

- Step 5: Deployment –> "Add", choose "Local or mountend folder" as Type and give a meaningful name like "localhost"

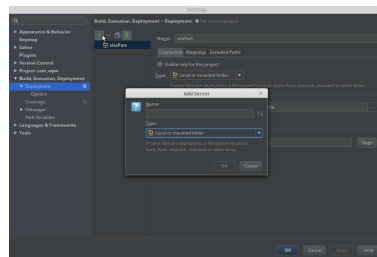


Figure 2.21: Open the PHPStorm settings

- Step 6: Folder = Path to your Joomla instance Web-server root Url : self-explanatory

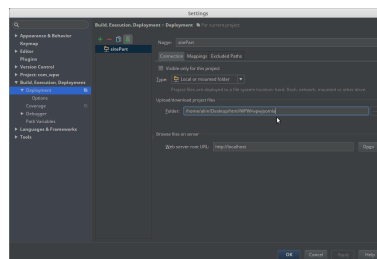


Figure 2.22: Add a new deployment

- **Step 7:** Local Path : Path to your local folder e.g.: `home/user/PhpStormProjects/com_ıcomponentnameı/site`  
Deployment Path : Path to the specific folder in your Joomla instance e.g.: `/components/-com_ıcomponentnameı`

By clicking on "add another mapping" you can add more mappings. Repeat the same steps for the other folders. ( Folder "admin" etc.).

**Please notice:** You cannot upload the same folder (or subfolders) to different sites at once. A separate deployment configuration has to be created for each of them and uploaded to each site one by one.

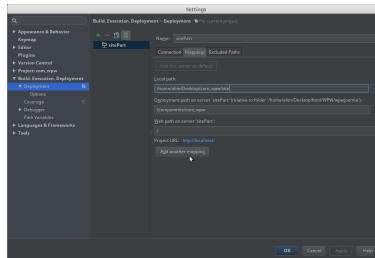


Figure 2.23: Add new mappings

- **Step 8:** Set the newly created deployment as your default deployment.

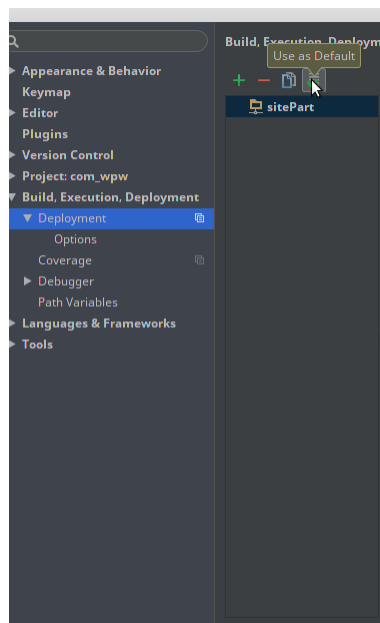


Figure 2.24: Set deployment as default

- Step 9: Set the sync options.

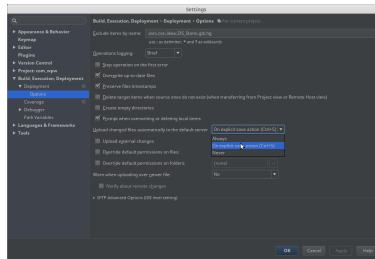


Figure 2.25: Set the sync options

## 2.4 Code Sniffer and Mess Detector

Now that we have checked out `lib.thm.core`, we can use the coding standards within it to set up PHP CodeSniffer.

PHP\_CodeSniffer is a PHP5 script that tokenises and “sniffs” PHP, JavaScript and CSS files to detect violations of a defined coding standard. It is an essential development tool that ensures your code remains clean and consistent. It can also help prevent some common semantic errors made by developers.<sup>5</sup>

PHP Mess Detector on the other hand searches for problems more abstract and more serious such as<sup>6</sup>:

- Possible bugs
- Suboptimal code
- Overcomplicated expressions
- Unused parameters, methods, properties

### Code Sniffer Installation

To install Code Sniffer for Joomla, follow the instructions on this page [https://docs.joomla.org/Joomla\\_CodeSniffer](https://docs.joomla.org/Joomla_CodeSniffer).

<sup>5</sup><http://pear.php.net/manual/en/package.php.php-codesniffer.intro.php> Stand September 22, 2014

<sup>6</sup><http://phpmd.org/> Stand September 27, 2014

### 2.4.1 Standards Directory Inclusion

First we will create symbolic links from the Joomla and THMJoomla folders to the folder which holds the existing standards. Open the windows shell as administrator and enter the following in the form **Command** **Link** **Target**.

#### Joomla Standard

**Command** `mklink /d`

**Link** `C:\xampp\php\pear\PHP\CodeSniffer\Standards\Joomla`

**Target** `C:\<System Specific Path>\lib.thm.core\coding_standards\Joomla`

#### iCampus Standard

**Command** `mklink /d`

**Link** `C:\xampp\php\pear\PHP\CodeSniffer\Standards\THMJoomla`

**Target** `C:\<System Specific Path>\lib.thm.core\coding_standards\THMJoomla`

### 2.4.2 PhpStorm

In order for Code Sniffer and Mess Detector to perform in PhpStorm they must first be integrated. This can be set as part of the default settings, meaning the settings will be applied to all projects, or in the project settings, where they would only valid for a specific project. The steps are exactly the same, but the entry point is slightly different. First access the settings using the menu item **File** and then selecting either **Default Settings...** or **Settings...** from the drop down menu.

#### PhpStorm Inclusion

To activate Code Sniffer click on **Languages & Frameworks**, **PHP** and then **Code Sniffer** or **Mess Detector**. In the text field next to “PHP Code Sniffer (phpcs) path” enter the path to phpcs.bat. In the standard installation this will be `C:\xampp\php\phpcs.bat`. After you have input the path, click on **Validate**. If the file is valid, the you will be shown a short text containing the version of the installed Code Sniffer. Although not specifically stated the steps for Mess Detector inclusion are completely analogous.

#### PhpStorm Configuration

Next we need to add **PHP\_CodeSniffer** to the inspections performed. With the settings still open, click on **Editor** then on **Inspections**, then in the list to the right **PHP**, and finally on **PHP Code Sniffer validation** to open the configuration settings for Code Sniffer.

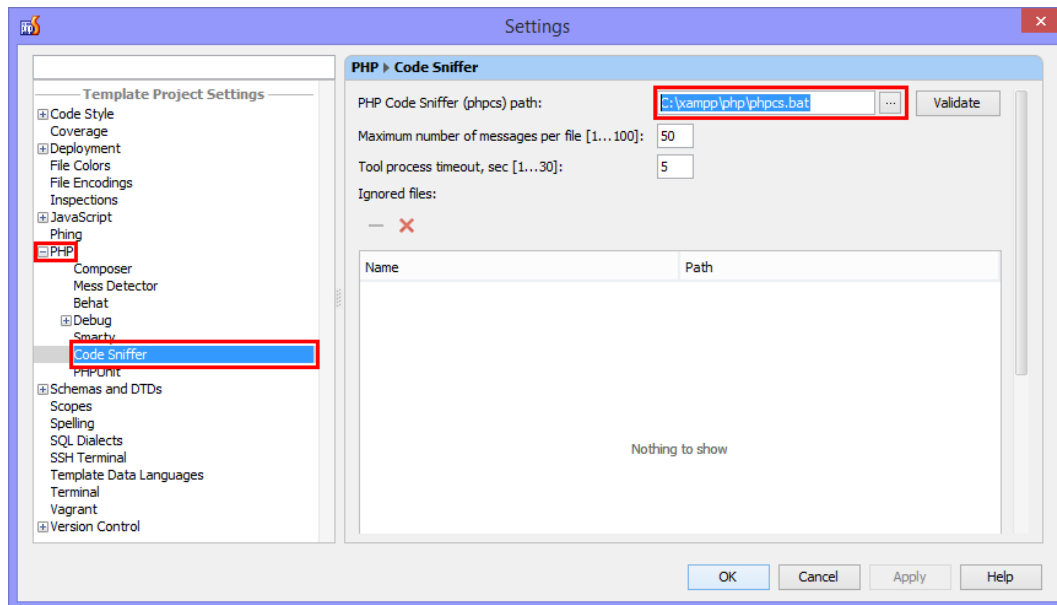


Figure 2.26: Code Sniffer - PhpStorm Inclusion

To activate the inspection we must first set the checkbox next to **PHP Code Sniffer validation**. This activation enables the further settings to the right which we see numbered with one through four in Figure 2.27.

1. **Inspection severity**: “indicates how seriously the code issues detected by the inspection impact the project and determines how the detected issues should be highlighted in the editor.”<sup>7</sup>
2. **Display severity**: defines how the infractions found are marked.
3. **Refresh**: updates the list of available coding standards. Should the desired standard not be found in the folder defined in the section on symbolic links, you can also manually search for the standard on your file system.
4. **Available standards**: a list of standards found. In iCampus we use the THM Joomla Standard which inherits or extends many of the definitions in the Joomla standard.

It is recommended that both inspection and display severity be set to errors to make the standard infractions more noticable. Settings take effect when the **Apply** button has been pressed.

Inspection settings for Mess Detector are also analogous here, with the distinction that one need not select a standard, instead selecting which rule sets should be applied.

<sup>7</sup><http://www.jetbrains.com/phpstorm/webhelp/configuring-inspection-severities.html> Stand September 22, 2014

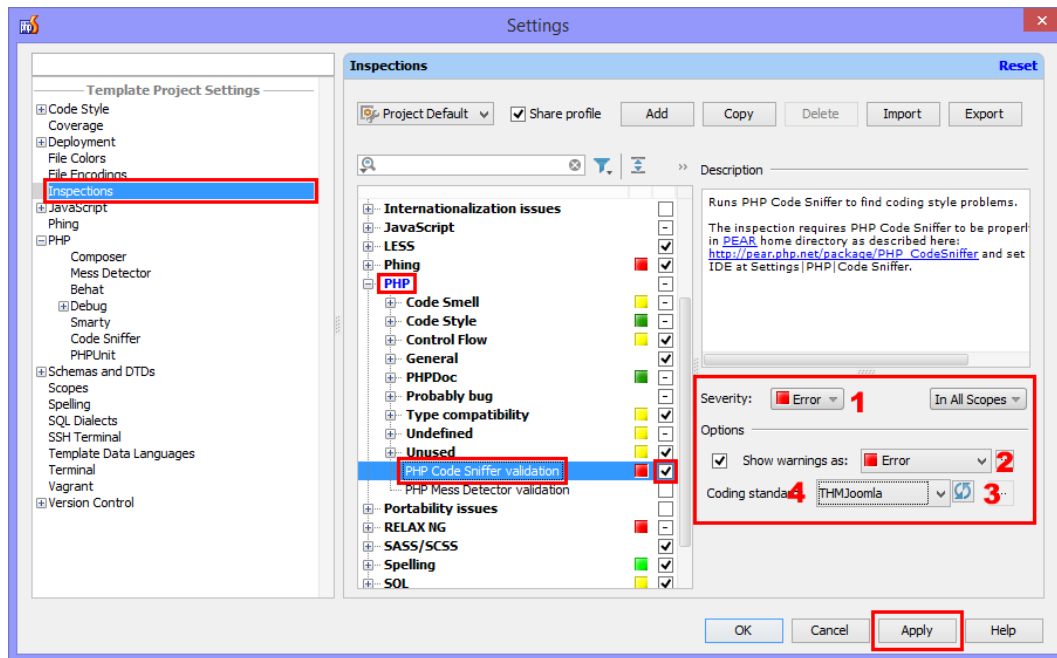


Figure 2.27: Code Sniffer - PhpStorm Configuration

## 2.5 SASS

In iCampus we strive to maintain a standardized appearance both within views of the same extension as well as between extensions developed by the iCampus Group as a whole. To this end we utilize SASS.

Sass is an extension of CSS that adds power and elegance to the basic language. It allows you to use variables, nested rules, mixins, inline imports, and more, all with a fully CSS-compatible syntax. Sass helps keep large stylesheets well-organized, and get small stylesheets up and running quickly, particularly with the help of the Compass style library.<sup>8</sup>

The above mentioned variables, nested rules, mixins, and inline imports allow the centralization of style structures and recurring themes, while still allowing for individualized style elements as required.

In iCampus we use the SCSS SASS syntax which closely resembles most .css you will have seen, and, stand alone, are actually valid CSS3 documents. This has as a consequence that our SASS files developed later will actually end with the extension .scss. For more information read [the SASS syntax explanation](#).

<sup>8</sup>[http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html](http://sass-lang.com/documentation/file.SASS_REFERENCE.html) Stand 27 September, 2014

### 2.5.1 Installation

Logically SASS must first be installed on the local system in order to function. This in turn requires the Ruby programming language, for which sass and compass are packages. While we need Ruby on the local system for SASS to function, we will not be using it to program, for this reason only the minimal requirements as pertain to SASS installation will be discussed.

Ruby can be installed by visiting [the ruby installer downloads page](#) and choosing the download that best suits your operating systems needs, as described in the right hand column.<sup>9</sup>

After you have downloaded and started the Ruby installation executable and accepted the license agreement you will be confronted with choices regarding the installation directory, support features, path variables, and file associations. We recommend using the default file path as it makes finding the installation easier, should any problems arise. You should only choose to install Td/Tk support if you know that you will be developing GUI applications in Ruby. The path variable is required in order for SASS to function. The association of .rb and .rbw files with this installation is recommended.

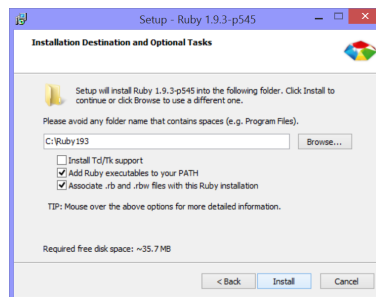


Figure 2.28: Ruby Installation

After ruby is installed we can begin to install SASS itself. Open the command line tool cmd, and enter the following command: `gem install sass`. Installation completes without an explicit notice but can be verified by entering: `sass -v`. The result should resemble the output of Figure 2.29.

Finally in order to extend the functionality of SASS, we install the Compass css authoring library package using the following command: `gem install compass`. Similar to SASS, no explicit message indicating the success of the installation is output, but can be verified with the command: `compass -v`.

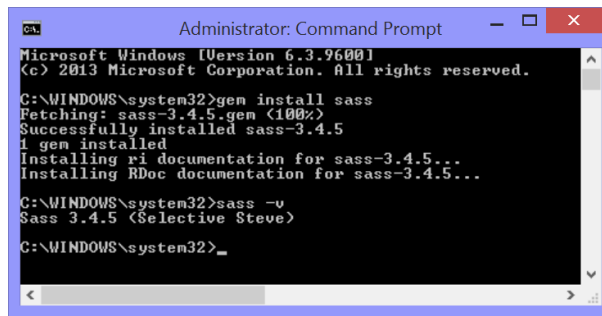
### 2.5.2 PhpStorm Configuration

In iCampus our SASS infrastructure, or SCSS in our case, are stored at iCampus level in `lib_thm_core\scss`, at project level in `<Project Name>\scss`, and lastly the compiled CSS files are stored in `<Project Name>\<Project Name>\media\css`.

---

<sup>9</sup>Stand 27 September, 2014





```
Administrator: Command Prompt
Microsoft Windows [Version 6.3.9600.1
(c) 2013 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>gem install sass
Fetching: sass-3.4.5.gem (100%)
Successfully installed sass-3.4.5
1 gem installed
Installing ri documentation for sass-3.4.5...
Installing RDoc documentation for sass-3.4.5...

C:\WINDOWS\system32>sass -v
Sass 3.4.5 (Selective Steve)

C:\WINDOWS\system32>
```

Figure 2.29: SASS Installation

The iCampus level SCSS files ensure standardized styles accross extensions. The project level styles allow for variance between individual extensions as required. To ensure that these files are not installed with the extension itself and thereby unnecessarily taking up system space these files are stored at root level in the project's repository.

Further complicating matters, SASS files are of two basic types, signified by the beginning of the file name. Templates are complete SASS files which are normally used to generate .css files, whereas partials contain reoccurring style information used by multiple style sheets. One of the many SASS conventions regards the naming of such files. Where templates have normal names, partials begin with a “\_” to signify them as such.

The configuration of the PhpStorm file watchers allows for all changes made to templates in the scss folder of the project to be compiled to the its css folder. Opening a template in PhpStorm will trigger a prompt asking if you would like to create a file watcher (Figure 2.30).

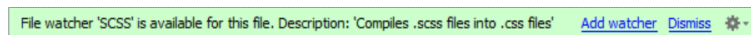


Figure 2.30: File Watcher Notification

To configure a file watcher either click on **Add watcher** in the notification, or navigate to **File > Settings** then click on **File Watchers**, the + button on the right hand side of the interface, and finally on **SCSS** in the selection box that appears.

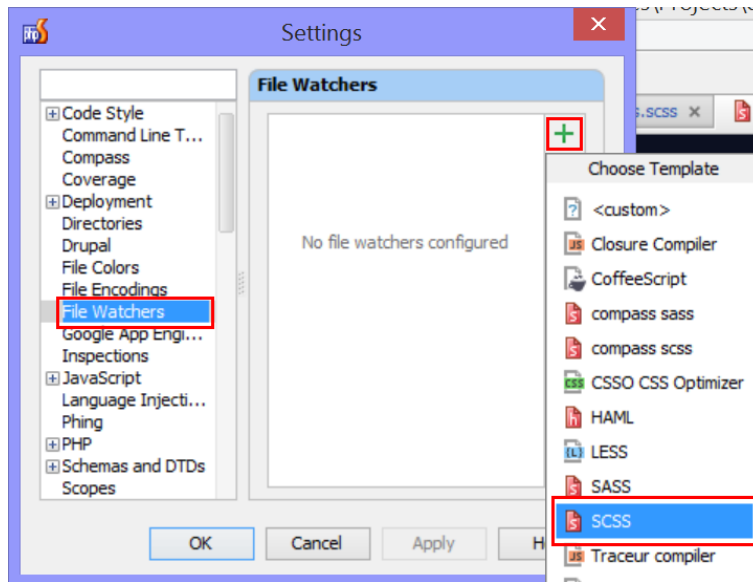


Figure 2.31: Add a New SCSS File Watcher

This brings us to the watcher edit interface seen in Figure 2.32. This will already have default values for all fields with the exception of **Program**. Fill out the fields as described below.

**Program:** <Ruby Installation Path>\bin\scss.bat

**Arguments:** --no-cache --update

\$FileName\$: \$FileParentDir\$/<Project Name>/media/css/\$FileNameWithoutExtension\$.css  
--sourcemap=none

**Output Paths:** \$FileParentDir\$/<Project Name>/media/css/\$FileNameWithoutExtension\$.css

Upon completion click “OK” to save the watcher. As necessary in the **Settings** interface put a check mark in the box in front of the watcher and click “Apply” for your changes to take effect.

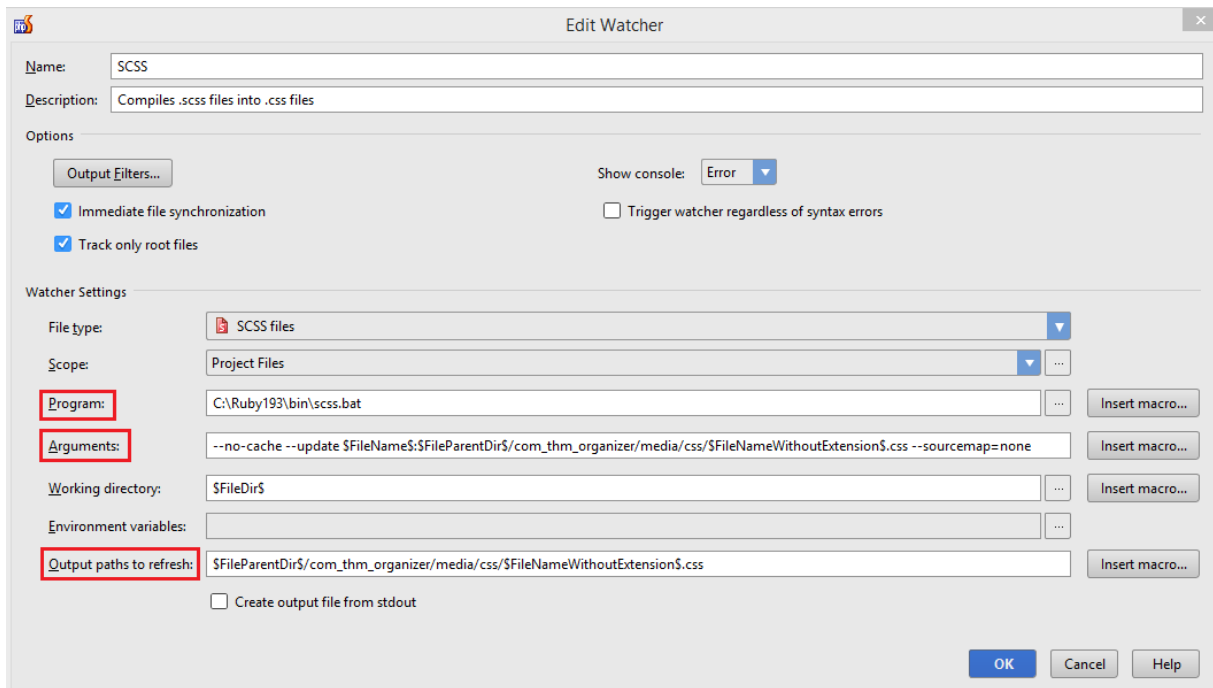


Figure 2.32: Edit File Watcher

*SASS version 3.4+ automatically generates <File Name>.css.map files when used as a plugin, as well as adds comments to the .css files which reference these. While this may at some point be used for debugging purposes, I would really like to turn it off. If anyone figures out how I would be very grateful. I could not get the method explained [here](#) to work.*

The inclusion of iCampus partials from `lib_thm_core` is done in the scss files themselves using `@import`. If your project repository is in the same directory as `lib_thm_core`, the imports would constructed as follows:

```
@import '../..../lib_thm_core/scss/<Partial Name without Underscore>';
```

*Here I would welcome any help in being able to add the directory with the partials to the SCSS path. This would reduce the import to the name of the partial to be imported.*

## 2.6 Joomla!

Almost all software developed by iCampus is for use within the context of the Joomla! content management system (CMS) used by the Department of Mathematics, Natural and Informatics (MNI) at the Central Hessen University of Applied Science.

Joomla is an award-winning content management system (CMS), which enables you to build Web sites and powerful online applications. Many aspects, including its ease-of-use and extensibility, have made Joomla the most popular Web site software available. Best of all, Joomla is an open source solution that is freely available to everyone.<sup>10</sup>

As of September 2014 iCampus is slightly between worlds as concerns Joomla!. The department's website currently still uses Joomla! version 2.5.x whereas the current Joomla! version as of 27 September, 2014 is 3.3.4.

For feature development and bugfixes for the department's website a dump of the site will be provided for you. Otherwise development should be performed in the context of Joomla! 3.x which can be downloaded from [the Joomla website](#). If you didn't deviate from the standard XAMPP installation you will want to extract the ZIP-Archive to the `C:\xampp\htdocs` directory.

At this point both the Apache and the MySQL services need to be running. To do so open the XAMPP control panel, Figure 2.2, and click the "Start" button next to both of these services.

### 2.6.1 Database Creation

For either of the development instances to function we first need to create databases for them to store their data in. To do this enter <http://localhost/phpmyadmin> in the address bar of your browser. This will open up the home page for your phpMyAdmin tool.

To create a database, click on the **Databases** tab, the interface for database management will then appear. The first item on the page will be "Create Database" under which will be an option for "Database name" and "Collation". Enter a name for your database and select the collation `utf8_general_ci` and then click **Create**. Then repeat as necessary for any further databases.

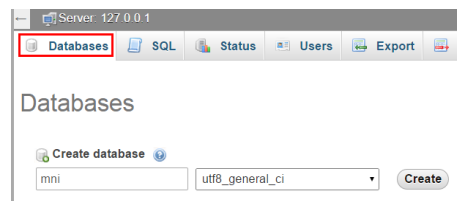


Figure 2.33: Database Creation

---

<sup>10</sup><http://www.joomla.org/about-joomla.html> Stand 27 September, 2014

## 2.6.2 MNI Site Development

First we will discuss the installation of the MNI backup. If you will only be developing for Joomla! 3.x you should skip ahead to the next subsection. First extract the backup and give it an URL-friendly name such as “mni”. Next place this folder in your web directory. If you followed the default installation steps this should be located at `C:\xampp\htdocs`.

Open a browser of your choosing and enter `localhost/<Your MNI Backup Folder>` in the address bar. If you renamed the backup “mni” this would then be `localhost/mni`. This will automatically start the Akeeba installer. The first page shown should look like Figure 2.34. As illustrated in the figure, we deviate from the recommended settings in one point “Display Errors”, since we changed it in Subsection 2.1.2. Should your settings not match those displayed, reconfigure your PHP settings to do so.

ANGIE – Akeeba Next Generation Installer Engine v 3.11.3

No idea what you are supposed to do? Don't panic! [Read the documentation page](#) [Watch the tutorial video](#)

Pre-installation Database Restoration Site Setup Finished

**IMPORTANT!** You are restoring on a server with a different PHP version than the one you used to back up your site. Your original site server's PHP version was 5.3.7+squeeze19 and your current PHP version is 5.5.15. Please note that different PHP versions may have differences which can cause your extensions (components, modules, plugins, libraries and templates) to not work properly. In these cases the restoration will complete without an error but your site may not display correctly or not load at all. Unfortunately, we cannot provide support for these issues. You will have to check that all of your extensions support PHP 5.5.15 before attempting to restore your site on this server.

**IMPORTANT!** You are restoring to a different site. We have detected that you are restoring to a different site than the one you backed up from. Some of your extensions, such as Admin Tools, may require to be reconfigured if they are using absolute paths or URLs. For more information please consult our Troubleshooter documentation.

### Pre-installation check

If any of these items is not supported (marked as No) then please take actions to correct them. Failure to do so could lead to your Joomla! installation not functioning correctly.

Setting	Current
PHP Version >= 5.2.4	Yes
Zlib Compression Support	Yes
XML Support	Yes
Database Support	Yes
MB Language is Default	Yes
MB String Overload Off	Yes
INI Parser Support	Yes
JSON Support	Yes
configuration.php Writeable	Yes

### Recommended settings

These settings are recommended for PHP in order to ensure full compatibility with Joomla!. However, Joomla! will still operate if your settings do not quite match the recommended configuration.

Setting	Recommended	Current
Safe Mode	Off	Off
Display Errors	Off	On
File Uploads	On	On
Magic Quotes Runtime	Off	Off
Magic Quotes GPC	Off	Off
Output Buffering	Off	Off
Session Auto Start	Off	Off
Native ZIP support	On	On

### Backup Information

This information was collected at the time of the backup. They represent the configuration of the server and site which was backed up. It is presented here for your reference and for easier debugging.

Setting	At Backup Time
Host name	www.mni.than.de
Backup date	2014-09-26 11:58:39 UTC
Akeeba Backup version	3.11.3
PHP version	5.3.7+squeeze19

[View README.html](#)

Click the button above to view the README.html file, generated at backup time, containing useful information about your backup.

### Site information

This information represents the configuration of the server you are restoring to (the server on which this installer is running)

Joomla! version	2.5.19
PHP version	5.5.15

Figure 2.34: Akeeba Pre-installation

Click “⇒ Next”. This brings us to the database restoration page seen in Figure 2.35.

ANGIE – Akeeba Next Generation Installer Engine v.3.11.3

← Previous Skip Restoration → Next

No idea what you are supposed to do? Don't panic! [Read the documentation page](#)

Pre-installation > **Database Restoration** > Site Setup > Finished

## Restoration of site's main database

**Connection information**

Database type:

Database server host name:

User name:

Password:

Database name:

**Advanced options**

With existing tables: Drop Backup

Database table name prefix:

☒ Suppress foreign key checks

☒ No auto value on zero

☐ Use REPLACE instead of INSERT

☐ Force UTF-8 collation on database

☐ Force UTF-8 collation on tables

**Fine tuning**

Do not change these settings unless you are requested to do so by our support or you *REALLY* know what you are doing.

Maximum execution time:

Throttle time (msec):

Figure 2.35: Akeeba Database Restoration

Here use the following values:

Database type	MySQL
Database server host name	localhost
User name	root
Password	(empty, unless set)
Database name	mni (unless named otherwise)
Maximum execution time	300

Clicking “⇒ Next” will trigger the restoration of the database. Then progress will then be displayed in a pop-up window. This may take up to several minutes dependent upon your computer. This brings us to the database restoration page seen in Figure 2.36.

ANGIE – Akeeba Next Generation Installer Engine v.3.11.3

No idea what you are supposed to do? Don't panic! [Read the documentation page](#)

Pre-installation > Database Restoration > **Site Setup** > Finished

### Site Parameters

Site name: Fachbereich MNI

Site e-mail address:

Site e-mail sender name:

Live site URL:

Cookie domain:

Cookie path:

☐ Override tmp and log paths

### FTP Layer Options

☐ Enable the FTP layer

Host name: 127.0.0.1

Port: 21

Username: jant89

Password:

Directory: [Browse...](#)

### Super User settings

Super User: jant89

E-mail: james.antrim@thm.de

Password:

Password (repeat):

### Directories fine-tuning

Site root: C:\xampp\htdocs\mni

Temporary directory: C:\xampp\htdocs\mni\tmp

Log directory: C:\xampp\htdocs\mni\log

Figure 2.36: Akeeba Site Setup

Here you need to choose a **Super User**. The available options are the users which have super user access to the live web site. Here it is not important which user you chose, but rather that you remember which user you chose. Also important is setting a local **Password** for the chosen user, this prevents the system from validating against the data available to the CAS server. Set a new local **Password** and repeat it in **Password (repeat)**.

When you click Clicking “⇒ Next” you will be directed to the final confirmation page. Here you will be notified that you need to delete the **installation** directory to complete the installation. Do so by clicking on the blue button. You will then automatically be redirected to the site’s frontend.

### 2.6.3 Joomla! 3.x Development

First abstract the archive file downloaded from [the Joomla! website](#) and give it a URL friendly name such as “j3”. Next place this folder in your web directory. If you followed the default installation steps this should be located at **C:\xampp\htdocs**.

Open a browser of your choosing and enter **localhost/<Your Joomla 3.x Folder>** in the address bar. If you renamed the backup “j3” this would then be **localhost/j3**. This will open the main configuration page as seen in Figure 2.37.

1 Configuration 2 Database 3 Overview

Select Language: English (United States) [Next](#)

### Main Configuration

Site Name \* Joomla 3.x Test Site  
Enter the name of your Joomla! site.

Admin Email \* james.antrim@mni.thm.de  
Enter an email address. This will be the email address of the Web site Super Administrator.

Description  
Enter a description of the overall Web site that is to be used by search engines. Generally, a maximum of 20 words is optimal.

Admin Username \* admin  
Set the username for your Super Administrator account.

Admin Password \* \*\*\*\*\*  
Set the password for your Super Administrator account and confirm it in the field below.

Confirm Admin Password \* \*\*\*\*\*

Site Offline: ☐ Yes ☒ No  
Set the site frontend offline when installation is completed. The site can be set online later on through the Global Configuration.

Figure 2.37: Joomla Main Configuration

Here you will need to enter the **Site Name**, **Admin Email**, **Admin Username**, **Admin Password**, and **Confirm Admin Password**. Click “⇒ Next” to move on to the database configuration.

1 Configuration 2 Database 3 Overview

Database Configuration [Previous](#) [Next](#)

Database Type \* MySQLi  
This is probably "MySQLi"

Host Name \* localhost  
This is usually "localhost"

Username \* root  
Either something as "root" or a username given by the host

Password  
For site security using a password for the database account is mandatory

Database Name \* j3  
Some hosts allow only a certain DB name per site. Use table prefix in this case for distinct Joomla! sites.

Table Prefix \* n1wmp\_  
Choose a table prefix or use the randomly generated. Ideally, three or four characters long, contain only alphanumeric characters, and MUST end in an underscore. Make sure that the prefix chosen is not used by other tables.

Old Database Process \* [Backup](#) [Remove](#)  
Any existing backup tables from former Joomla! installations will be replaced

Figure 2.38: Joomla Database Configuration

Use the following values to complete the form, then click “⇒ Next” to continue on to the finalization.

Database type	MySQLi
Host Name	localhost
Username	root
Password	(empty, unless set)
Database name	j3 (unless named otherwise)



The screenshot shows the Joomla! installation process. At the top, there are three tabs: '1 Configuration', '2 Database', and '3 Overview', with '3 Overview' being the active tab. Below the tabs, the 'Finalisation' section is visible, featuring a 'Previous' button and an 'Install' button. Under 'Install Sample Data', the 'None (Required for basic native multilingual site creation)' option is selected. Other options include 'Blog English (GB) Sample Data', 'Brochure English (GB) Sample Data', 'Default English (GB) Sample Data', 'Learn Joomla English (GB) Sample Data', and 'Test English (GB) Sample Data'. A note states: 'Installing sample data is strongly recommended for beginners. This will install sample content that is included in the Joomla! installation package.' Below this, the 'Overview' section shows 'Email Configuration' with 'Yes' and 'No' buttons. The 'No' button is selected, and a text field contains the email address 'james.andrim@nnn.thm.de'. A note below the text field says: 'Send configuration settings to james.andrim@nnn.thm.de by email after installation.'

Figure 2.39: Joomla Finalization

Here you can choose whether or not you wish to install sample data. Leave “None” selected and click “⇒ Install” to complete the installation. The installation will then show the progress of the database construction. When this is finished you will be told that Joomla! is installed. You should now press the orange button “Remove installation folder” to complete the installation. You can choose to visit the frontend or backend of the site by clicking the appropriate button.

# Chapter 3

## Tool Guide

### 3.1 Git

#### Revert to the previous commit

From the repository directory in Git Bash or the PhpStorm Terminal enter:

```
git reset --hard HEAD
```

*Warning! This will erase any and all uncommitted changes!*

#### Create a new local branch

#### Add a local branch to remote

From the repository directory in Git Bash or the PhpStorm Terminal enter:

```
git push -u origin <Branch Name>
```

### 3.2 Gerrit

Gerrit is a web based code review system, facilitating online code reviews for projects using the Git version control system.

Gerrit makes reviews easier by showing changes in a side-by-side display, and allowing inline comments to be added by any reviewer.

### 3.2.1 Workflow

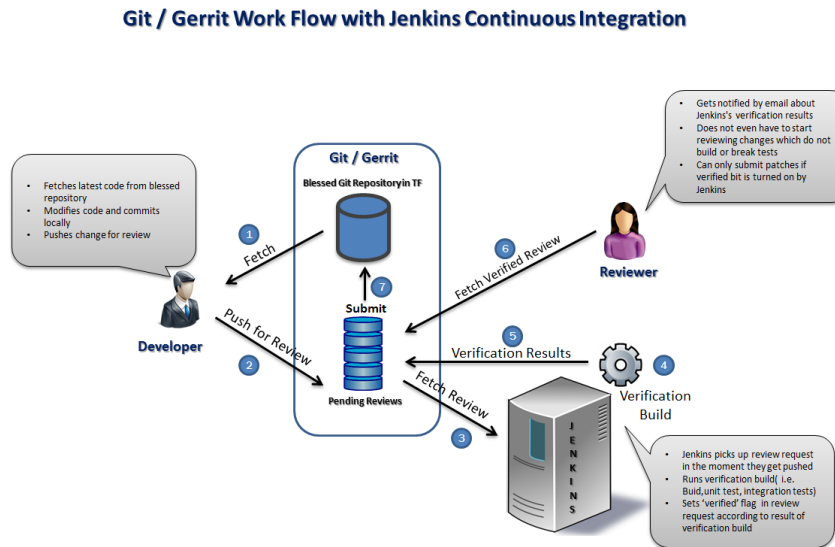


Figure 3.1: Git Gerrit Jenkins Workflow

### 3.2.2 Cloning repository

In order to clone a repository to your local computer look at section [2.3.4](#) <sup>1</sup>.

### 3.2.3 Push to repository

After working on the project we need to push our changes back to gerrit. To do so just rightclick the project and select Git – Commit Directory...

---

<sup>1</sup>[2.3.4](#) THM Core Library

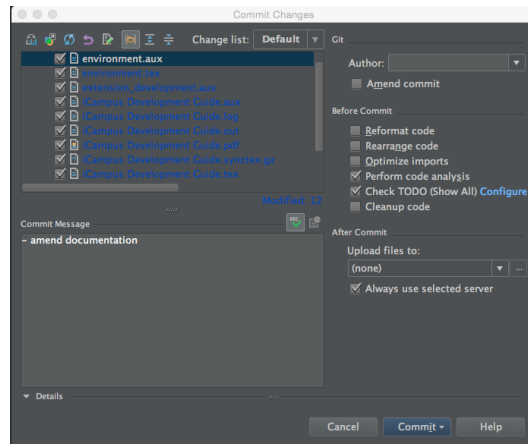


Figure 3.2: Gerrit Commit Directory

**Please watch out!** If you want to recommit a modification depending the same change, be sure to check the Amend commit Checkbox. This will not lead to a new Changeset but in an update of your precommitted change. Do not delete old Commit-Message. Just amend the message.

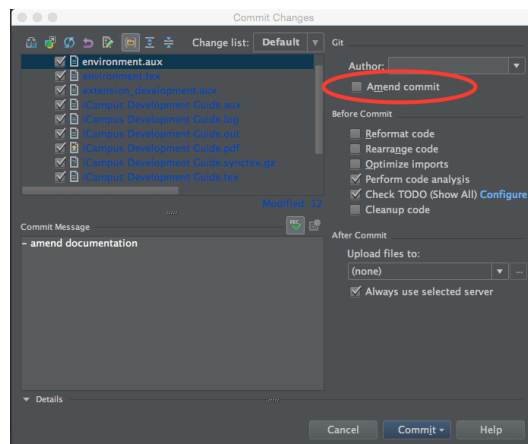


Figure 3.3: Gerrit Amend Commit

After we committed the change with a useful comment, we need to push it to Gerrit. In spite we have the latest version of the Gerrit plugin in PhpStorm we just need to rightclick the project and select Git – Push and check "Push to Gerrit". It is possible to declare reviewers who will be informed about the push.

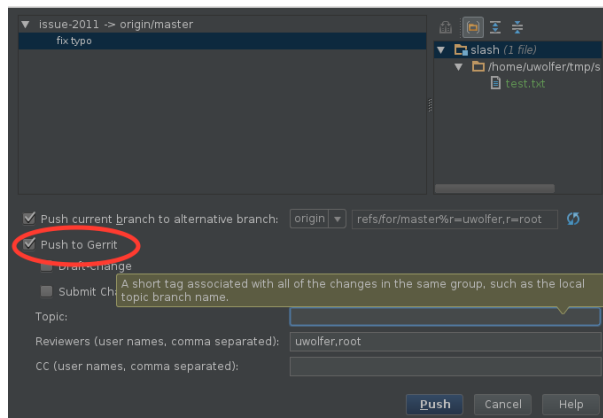


Figure 3.4: Gerrit Push New

If the Plugin is not at its latest version we have to push to a specific branch. Please check the box at "Push current branch to alternate branch" and type the following in the textbox: "refs/-for/master". We should not be able to push to any other branch except we are the projectowner.

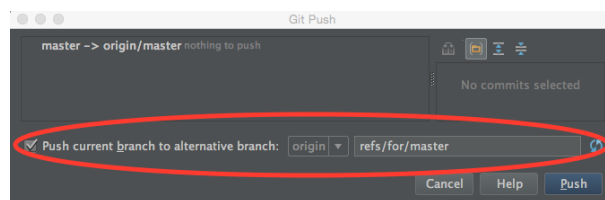


Figure 3.5: Gerrit Push Old

Afterwards we should be able to see the change in the gerrit window of our PhpStorm installation.

### 3.2.4 Review change

The Gerrit Tool Window<sup>2</sup> will show us all changes for the project we are working on. We have access on different actions like review, submit...

<sup>2</sup>[2.3.5 Tool Window Activation](#)

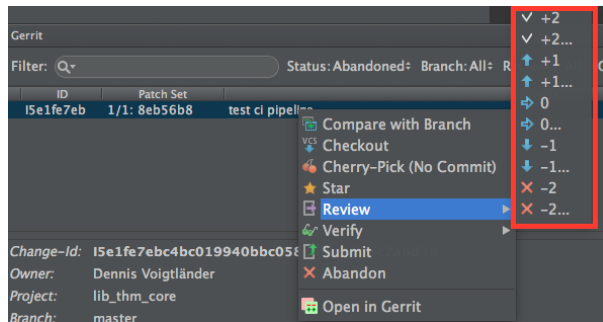


Figure 3.6: Gerrit Review Scores

A change has to have a minimum Review-Score of +2 in order to be merged into master. The change also has to be verified by Jenkins. Only the projectowner is authorized to merge the change into master. He can do it by clicking on the change and choosing "Submit".

### 3.2.5 Submit/Abandon change

Submitting a change in Gerrit means that an attached change gets merged into master (only possible for projectowner). Therefore a release will be created. The projectowner can right-click a change and choose "Submit" if the change should be released as a patch. It will only succeed if the change is reviewed with a minimum score of +2 and if it's verified +1 by Jenkins. If the change should be released as a feature, the projectowner needs to visit the Webinterface<sup>3</sup> of Gerrit and type in the topic of the change "feature" before he submits (until now - will also be able in IDE in near future).

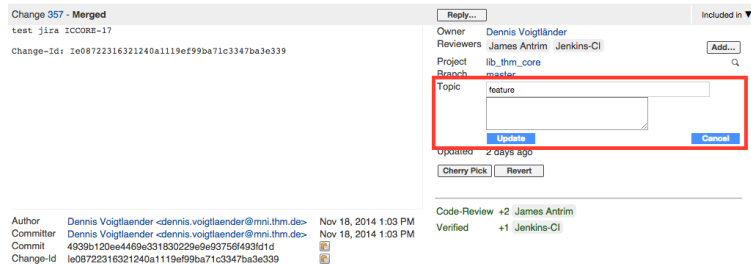


Figure 3.7: Gerrit Feature

If nothing is declared in the textbox topic, the change will be released as a patch.

<sup>3</sup><http://www.gerrit.mni.thm.de/gerrit>

### 3.2.6 FAQ and Help

#### If you haven't rebased and HEAD is lost

If you committed without rebasing, your local HEAD is x-commits behind the HEAD of the Server. To fix this, you need to open up the Terminal located in the taskbar on bottom left. Once the terminal is open, type 'git reset HEAD~x' (where x means the number of commits you're behind the HEAD of the Server). In the following example, the local HEAD is 1 commit behind the server HEAD.

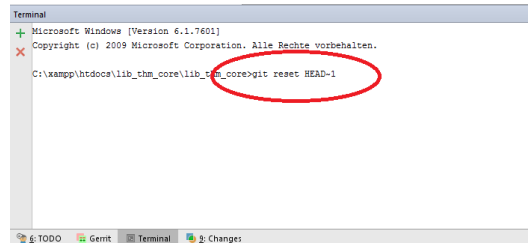


Figure 3.8: Terminal

Now you have to press the blue down arrow with VCS label on the top.

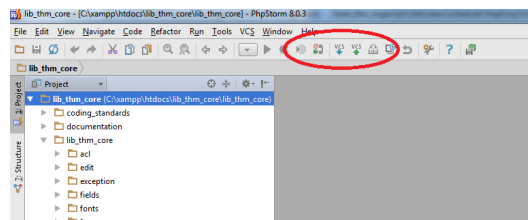


Figure 3.9: Terminal

After that, choose 'Rebase' in the next window.

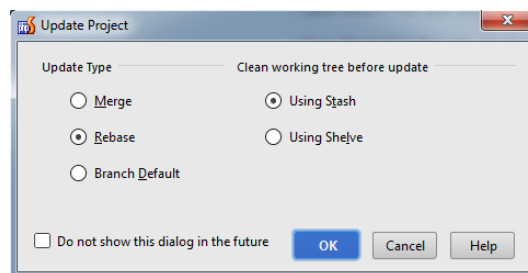


Figure 3.10: Terminal

These steps should be done after every commit. Now that you've synchronized the HEAD, you can commit changes as described in ??, Section ??.

# Chapter 4

## Development Guide

### 4.1 Workflow

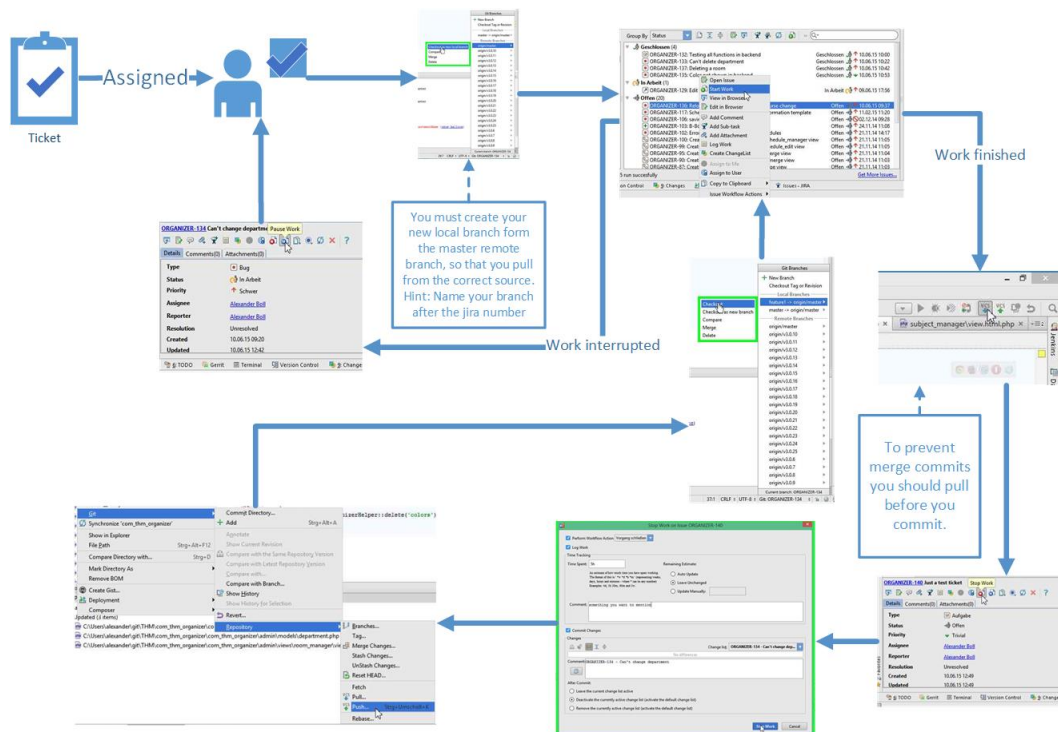


Figure 4.1: Gerrit Workflow



## 4.2 Developing

### 4.2.1 Advanced PHP Tips

#### String Equals Zero In PHP

<sup>1</sup>Due to the weakly-typed nature of PHP you can do some odd things, some of which are good, and some of which will enable you to shoot yourself in the foot. Care if you compare string with integer values. Take the following little snippet.

Maybe you expect that the in-built function `var_dump()` outputs `bool(false)` because 0 is not equal 'attributes' but instead the result is `bool(true)`! When you compare an integer and string, PHP converts the string to an integer. The integer of the string 'attributes' is 0. So `var_dump(0 == 0)` outputs `bool(true)`.

## 4.3 Integration

### 4.3.1 Reviewing

As a reviewer one has the responsibility of ensuring code quality. The Gerrit plugin offers many features which make this process easier. To open the Gerrit plugin's interface click on the icon on the bottom edge of the PHPStorm interface, as in [Figure 4.2](#).

---

<sup>1</sup>Original text from: <http://www.hashbangcode.com/blog/string-equals-zero-php>

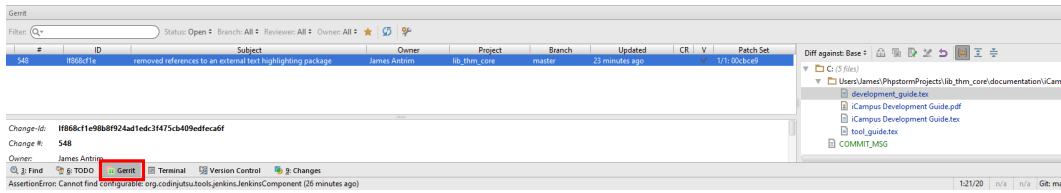


Figure 4.2: Gerrit Plugin Interface

In this figure we see a previous commit already highlighted. This commit has already been examined by Jenkins and has been deemed to be of acceptable quality, this is evinced by the checkmark in the 'v' column. Jenkins can only perform automated tests and checks, which are good for problems which can be detected in this manner. It cannot find every error, and it is at this point which the review system seeks to iron out, letting reviewers examine code before it is permanently made of a release.

Because it is highlighted, the files changed by this commit are listed on the right hand side. To see the changes to individual files right click on the file in this list and select **Show Diff**.

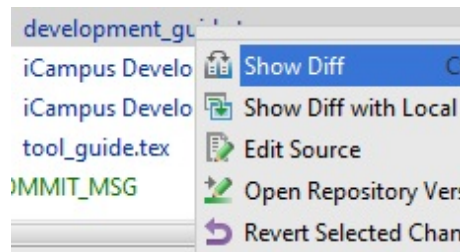


Figure 4.3: File Review Interface

This opens a further interface where the local contents are then compared with those in the change. A reviewer can then examine changes line by line to ensure the desired code quality has been reached. At this point multiple options open, but only the most important will be described.

## Approval

Should the code be of an acceptable quality the reviewer can merge the commit with the master branch by right clicking on the list entry, opening the context menu (Figure 4.4). Then the reviewer can 'review' the commit by hovering over the word **Review** to the right a sub menu opens in which he selects +2. This tells Gerrit that the code has been reviewed and approved. By right-clicking again on the entry and left-clicking on submit the commit will then be merged into the master branch.

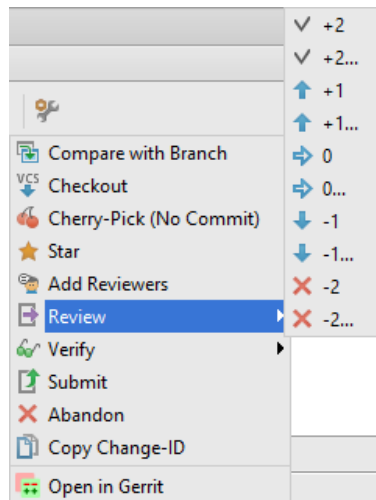


Figure 4.4: Commit Context Interface

## Rejection

If the code needs improvement it can either be rejected outright or fixed. Outright rejection is performed over the same context menu as approval, only instead of +2, the reviewer then selects −2 or −2... The second option allows for a rejection message, which can be useful for stating the reason why the commit was rejected.

The other option is to correct or improve the commit. To do so, the reviewer opens the commit context menu and selects **Checkout**. This creates a new local branch with the same stand as the commit to be corrected or improved. In the bottom right hand corner of the PhpStorm interface the new branch is now displayed. Click on the branch name to open the branch-interface. Select master and in the context menu that opens select **Checkout**. This sets your repository to the stand of the current master branch. Next reopen the branch-interface and click on the newly created local branch. In the menu that now opens select **Merge**. This merges that changes from the new branch onto the stand of the master.

Now that you have the commit's changes in your master branch proceed to correct and improve the changed files as appropriate. When you have completed your changes append them to the previous commit by placing the checkmark in the **Append Commit** box and performing the commit and push as you normally would.

## 4.4 Testing

<sup>2</sup>The black box in Figure 4.5 shows the repository structure with the extension and test folder structure. The tests are separated by unit and GUI tests. Integration tests are also stored in the unit-tests folder. Should it be necessary, they could also be in a separate folder. The green box in Figure 4.5 shows the unit-tests folder in more detail. This folder is structured into a general part (*schema* and *stubs*) and the two folders *admin* and *site*. The folder *admin* contains the tests for the back-end of the extension. And also the test suite (phpunit.xml) and the bootstrap (component\_bootstrap.php) file. The *site* folder is structured like the *admin* but has the unit tests for the front-end of the extension in it. Both folders have to include a bootstrap file because some Joomla path constants are different for the back-end and the front-end.

The folder *schema* includes the database schema file (ddl.sql). The database schema is written in SQLite syntax<sup>3</sup>. This file is used by the tests to create the SQLite database tables.

Directly under the *stubs* folder files are located that contain data for the tests. For instance, a long text or a JSON string is stored, to keep the tests readable. The folder *database* contains files in which the database entries are stored. The entries are loaded by the tests and stored into the SQLite database tables.

The blue box in Figure 4.5 shows the gui-tests folder in more detail. Like the unit-tests folder, the gui-tests folder is divided into a back-end (admin) and a front-end (site) part. The *Pages* folder contains classes that implements the Page Object pattern. The bootstrap file (component\_bootstrap.php) is the same for the back-end and front-end. This file registers the classes in the *Pages* folder so that the GUI tests can use them without explicitly include them.

To be able to execute the tests, they must be copied to the *tests* folder in an Joomla instance. Figure 4.6 shows the file structure of the *tests* folder in an Joomla instance. This folder also includes the basic TestCase classes (tests/core/case), Hard-Coded Mock Objects (tests/core/mock) for general Joomla objects (JApplication, JConfig, JDocument, JLanguage, etc.) and a reflection helper class (tests/core/reflection) to get private and protected values in a class by reflection<sup>4</sup>.

In addition, the folder contains Page Objects classes (tests/Pages) for all GUI tests because they represent webpages, which all extensions can use, like the login page, the control panel in the back-end and more. Furthermore there are iCampus Page Objects classes (test/Pages/icampus) that provide methods for general tasks, like add/edit/delete an item in a manager view in the back-end. The last folder (tests/SeleniumClient) contains the PHP-SeleniumClient which allows to interact with the Selenium server. The file bootstrapJ3.php includes the Joomla framework and registers the TestCase classes. This file is again included by the component\_bootstrap.php file (tests/com\_thm\_organizer/unit-tests/admin & site). The bootstrapSelenium.php file makes the general Page Object classes available for the GUI tests and includes the JoomlaWebdriverTestCase.php. This file is the superclass of the GUI test classes and initialises a WebDriver object (SeleniumClient) using the configuration in the seleniumConfig.php file. All the general folders in the tests folder except the com\_thm\_organizer folder are stored in the joomla.ci repository (<sup>5</sup>) in GITBlit. Jenkins and the developers just have to copy the general folders from the CI repository and the tests from the extensions repository to an instance of Joomla and can immediately execute the tests. The requirements to execute the tests are listed below. Almost always the requirements are met on the

---

<sup>2</sup>Source: Rost, Wolf (2014): Software Tests in Agile Web-CMS Development. TH-Mittelhessen.

<sup>3</sup>See <http://www.sqlite.org/lang.html>, stand 31.08.2014

<sup>4</sup>See <http://php.net/manual/de/book.reflection.php>, stand 31.08.2014

<sup>5</sup>joomla.ci repository: [http://dev.mni.thm.de/gitblit/summary/ci!joomla\\_ci.git](http://dev.mni.thm.de/gitblit/summary/ci!joomla_ci.git)

machines of developers.

- PHPUnit 4.0 (minimum)<sup>6</sup>
- PHP 5.4 (Joomla also requires PHP)<sup>7</sup>
- An Joomla 3 instance with the installed extensions.
- SQLite3 (Is enabled by default.)

## 4.5 Debugging

To debug Joomla and PHP in general you need three things:

- xdebug configuration in the php.ini (see 2.1.2)
- PHPStorm installed (see 2.3)
- If you use Firefox you have to install the addon "The easiest Xdebug". If you use Google Chrome you have to install the addon "Xdebug helper".

If you have all these requirements you can activate the addon in your browser, set a break point in your php file and start listening for php debug connections in PHPStorm.

---

<sup>6</sup>See <https://phpunit.de/manual/current/en/installation.html>, stand 19.01.2014

<sup>7</sup>See <http://www.joomla.org/technical-requirements.html>, stand 01.09.2014

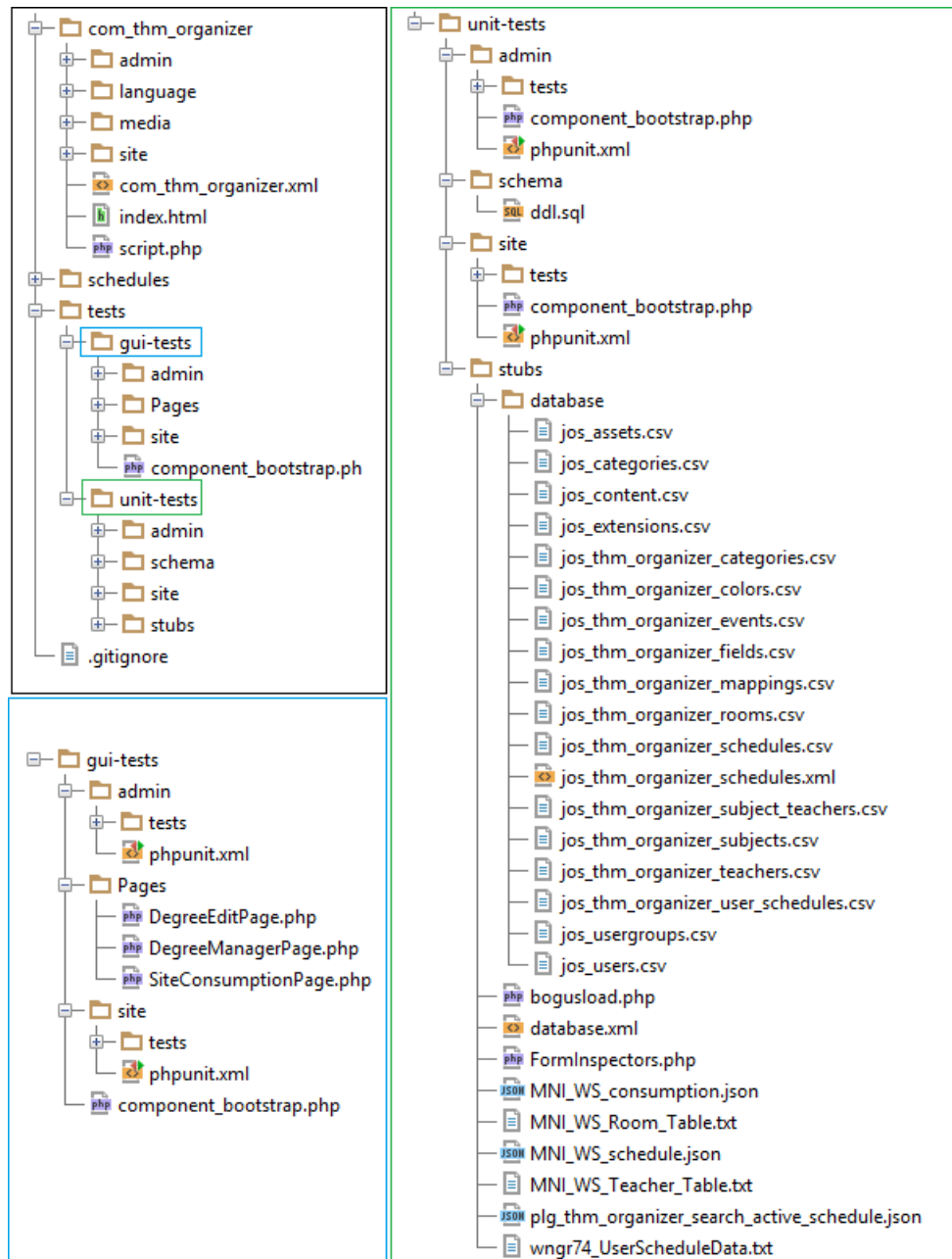


Figure 4.5: Repository file structure of the THM Organizer component. The black box gives an overview of the file structure. The blue box shows the gui-tests folder in detail. The green box shows the unit-tests folder in detail.(Source: Rost, Wolf (2014): Software Tests in Agile Web-CMS Development. TH-Mittelhessen.)

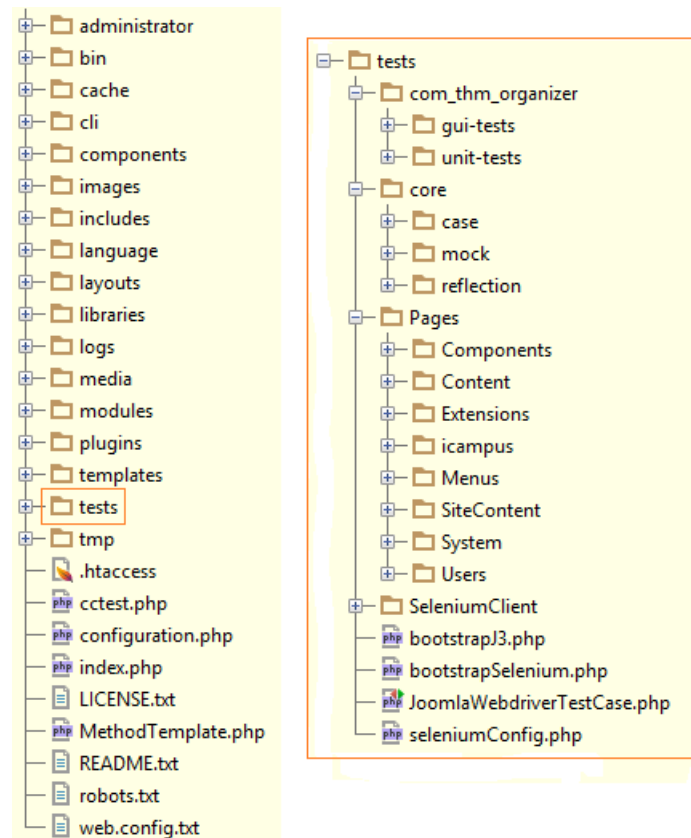


Figure 4.6: Test folder structure of an Joomla instance. (Source: Rost, Wolf (2014): Software Tests in Agile Web-CMS Development. TH-Mittelhessen.)

# Appendix A

## How-tos

### A.1 How to change environment variables in Windows

To view or change environment variables:<sup>1</sup>

1. Right-click My Computer, and then click Properties.
2. Click the Advanced tab.
3. Click Environment variables.
4. Click one the following options, for either a user or a system variable:  
Click New to add a new variable name and value.  
Click an existing variable, and then click Edit to change its name or value.  
Click an existing variable, and then click Delete to remove it.

---

<sup>1</sup>For the original documentation see <http://support.microsoft.com/kb/310519/en-us>