```
/**
 * Rock ChMS
 * Developer
 * Handbook
 */
```

OBSOLETE – moved to wiki

Version: 0.1.35

Last updated: 11/8/2012

spark
developement network

Table of Contents

Keeping Developers Happy

Developers are fickle and that’s OK. They have good and bad habits, and have opinions on how things should be done. Unfortunately if left untamed, this can lead to a state of unhappiness for everyone. For example, attempting to perform diffs against code whose tab-spacing changes from version to version makes developers unhappy.

This handbook will contain the **rules and regulations** which are intended to keep everyone happy and it will also explain some of the environment setup needed to get started with developing RockChMS.

Setting Up Your Development Environment

Getting Git

SmartGit is a Git client for Windows, Mac OS X and Linux. It provides a GUI to the Git command-line interface. We recommend SmartGit because it's free for non-commercial use.

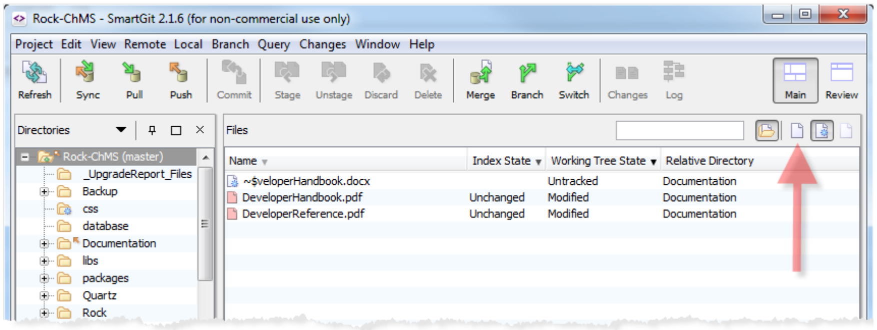
1. Download and install "**Git for Windows**" from here http://code.google.com/p/msysgit/ with all the default settings.
2. Download and install SmartGit.
   a. Follow this to install Git http://help.github.com/win-set-up-git/
3. Run SmartGit to complete SmartGit setup using the defaults except where noted below:
   a. For the **Type of Usage** step you will probably want to select the third option, "Non-Commercial Use Only" as long as you're only working on the non-commercial Rock-ChMS project.
   b. For the **Git & HG Executables step,** verify that the Git Executable path is filled in with the path to where you installed Git from step 1.
   c. For the **User Information** step, use your GitHub account name and email.
   d. For the **Hosting Providers** step, select GitHub as your main hosting provider.
   e. For the Master Passwords… step, we **do not** recommend choosing the "Don't use a master password" option.
   f. In the Welcome to SmartGit window, if asked "What do you want to do?" just cancel/close that window and proceed to the next section.

Getting RockChMS

1. Locate the project's Git repository by visiting https://github.com/SparkDevNetwork/Rock-ChMS.
2. Find the project's repository read-only URL address (it will be something like https://github.com/SparkDevNetwork/Rock-ChMS.git) if you're on the Rock developer core team you can use your read-write Url or if you've forked the Rock-ChMS repo, you can use the read-write Url to your forked repo.

3. In SmartGit, go to “Project” » “Clone” (or choose “clone existing repository”) and use the default settings except where noted here:
    a. In the **Repository Url** field, enter “https://github.com/SparkDevNetwork/Rock-ChMS.git”
    b. In the **Local Directory** step, locate or create a folder on your computer where you’ll store your Git Working Tree, e.g., C:\projects\Rock-ChMS.

NOTE: Most of us find it’s best to unselect the “show unchanged files” option (Crtl+1) in SmartGit.



Name	Index State	Working Tree State	Relative Directory
~$veloperHandbook.docx		Untracked	Documentation
DeveloperHandbook.pdf	Unchanged	Modified	Documentation
DeveloperReference.pdf	Unchanged	Modified	Documentation

Getting Started

Database

You’ll need to create an empty “RockChMS” database on your local system with a default user (owner) of “RockUser”. When you run “Update-Database” (described below), the EF framework will create all the needed tables and data.

> Note: Make sure your SQL Server properties Security’s Server Authentication is set to “SQL Server and Windows Authentication mode”. You’ll also need to enable TCP/IP and Named Pipes in your SQL Server Client Protocols (if you have not already done this before).

Web Config

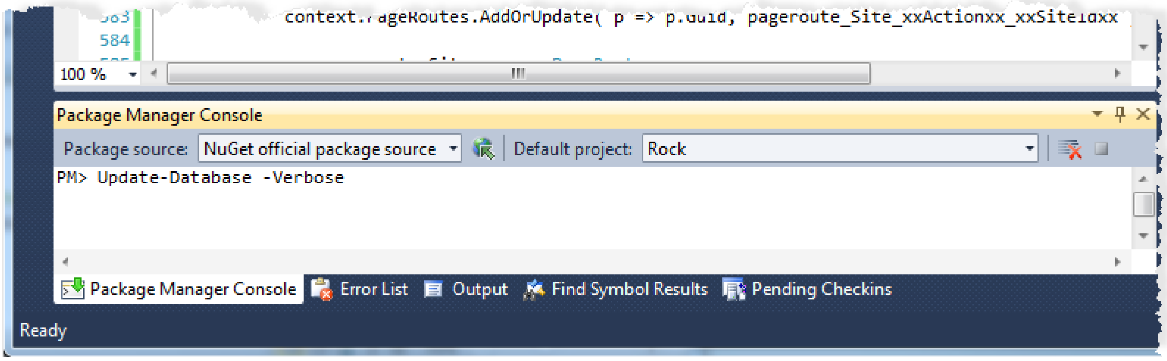
If it does not exist, create an empty web.connectionString.config file under your RockWeb project. Add with something like:

```
<connectionStrings>
    <add name="RockContext" connectionString="Data Source=localhost;Initial Catalog=RockChMS; User Id=RockUser; password=<PASSWORD>;MultipleActiveResultSets=true" providerName="System.Data.SqlClient" />
</connectionStrings>
```

Replace <Password> with the one you created in the Database step above.

First Time – Creating and Seeding the Database

Start Visual Studio and open the Package Manager Console window and run "Update-Database" (you can pass the –Verbose flag if you wish) as shown below. *Make sure you're Default project is set to Rock as shown here in this screenshot.*



```
context.PageRoutes.AddOrUpdate( p => p.Guid, pageroute_Site_xxActionxx_xxSiteIdxx
```

Package source: NuGet official package source

Default project: Rock

```
PM> Update-Database -Verbose
```

Once you do this, your RockChMS database should have all the required tables and initial seed data required to be able to run Rock as the Administrator user.

Press F5 to launch Rock in Debug mode and then login to rock using the default administrator login (Admin) and password (Admin).

Coding Standards

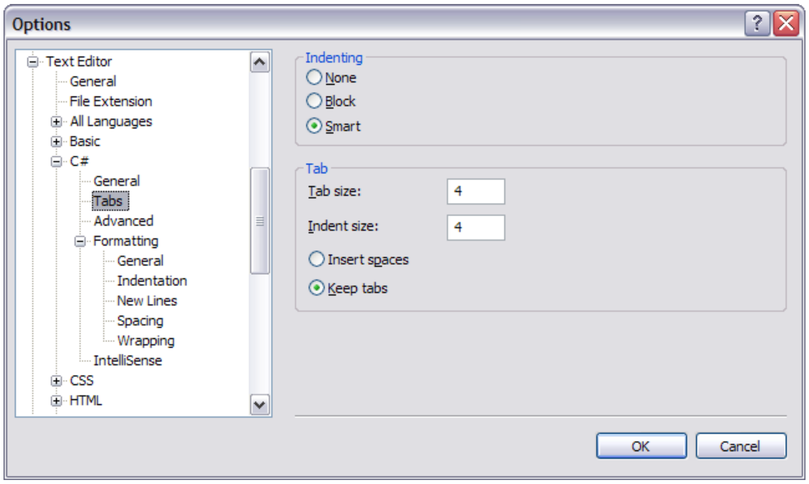
The following are the adopted coding standards:

- First and foremost, follow the established patterns you see in the existing code base. Don't invent your own patterns.

- See http://www.dotnetspider.com/tutorials/CodingStandards.doc

- Follow the Naming Conventions and Standards outlined by items 1 through 11 in section 7.

- Follow the Indentation and Spacing outlined in section 8. Use Spacing Settings for Visual Studio. Under Options, Text Editor, C#, Spacing checkbox the following (you may need to check the "Show all settings" checkbox in the lower left corner if you don't see these settings):



Indenting

○ None

○ Block

● Smart

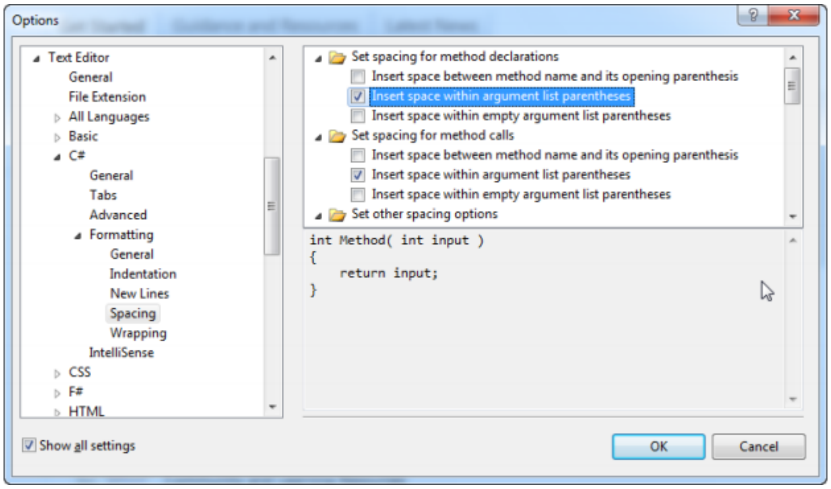
Tab

Tab size: 4

Indent size: 4

○ Insert spaces

● Keep tabs



Set spacing for method declarations

□ Insert space between method name and its opening parenthesis

☑ Insert space within argument list parentheses

□ Insert space within empty argument list parentheses

Set spacing for method calls

□ Insert space between method name and its opening parenthesis

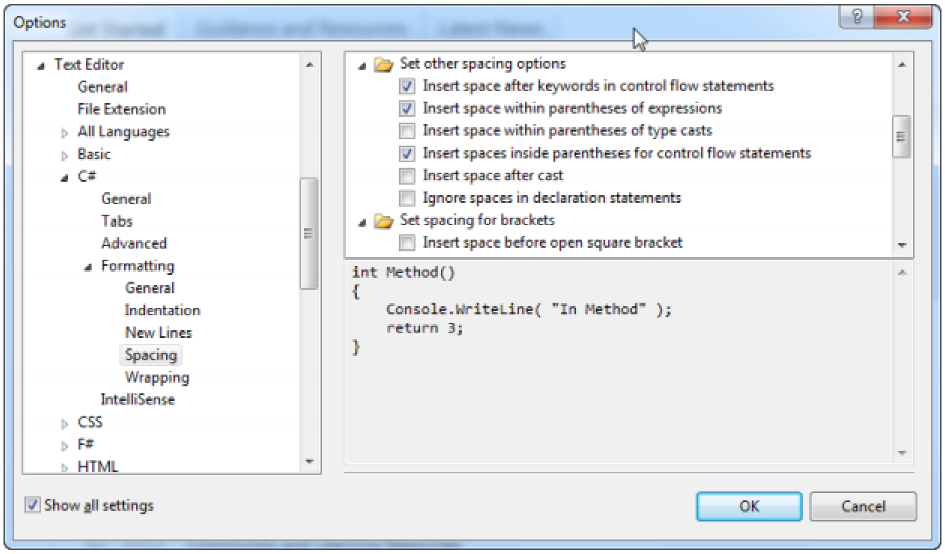
☑ Insert space within argument list parentheses

□ Insert space within empty argument list parentheses

Set other spacing options

```
int Method( int input )
{
    return input;
}
```

☑ Show all settings



Set other spacing options

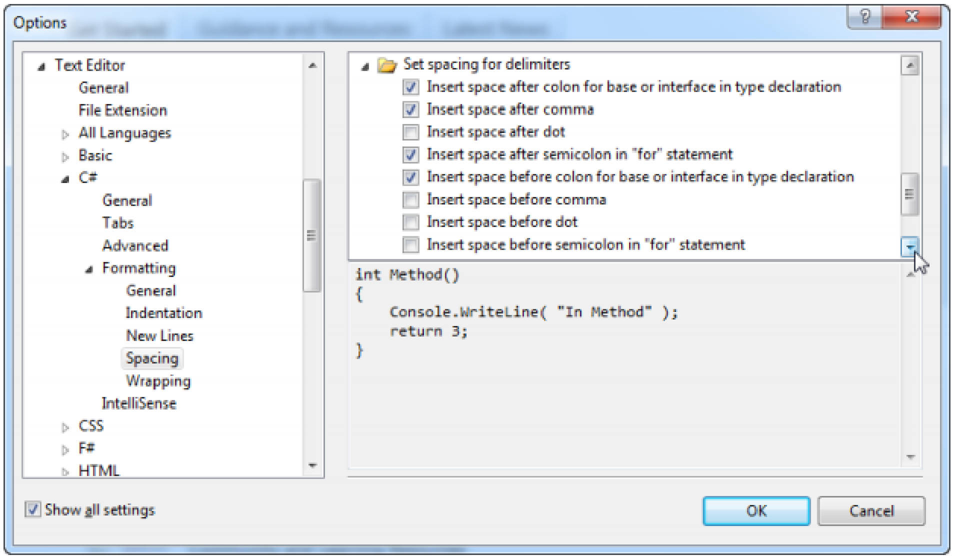
- ☑ Insert space after keywords in control flow statements
- ☑ Insert space within parentheses of expressions
- ☐ Insert space within parentheses of type casts
- ☑ Insert spaces inside parentheses for control flow statements
- ☐ Insert space after cast
- ☐ Ignore spaces in declaration statements

Set spacing for brackets

- ☐ Insert space before open square bracket

```
int Method()
{
    Console.WriteLine( "In Method" );
    return 3;
}
```

☑ Show all settings



Set spacing for delimiters

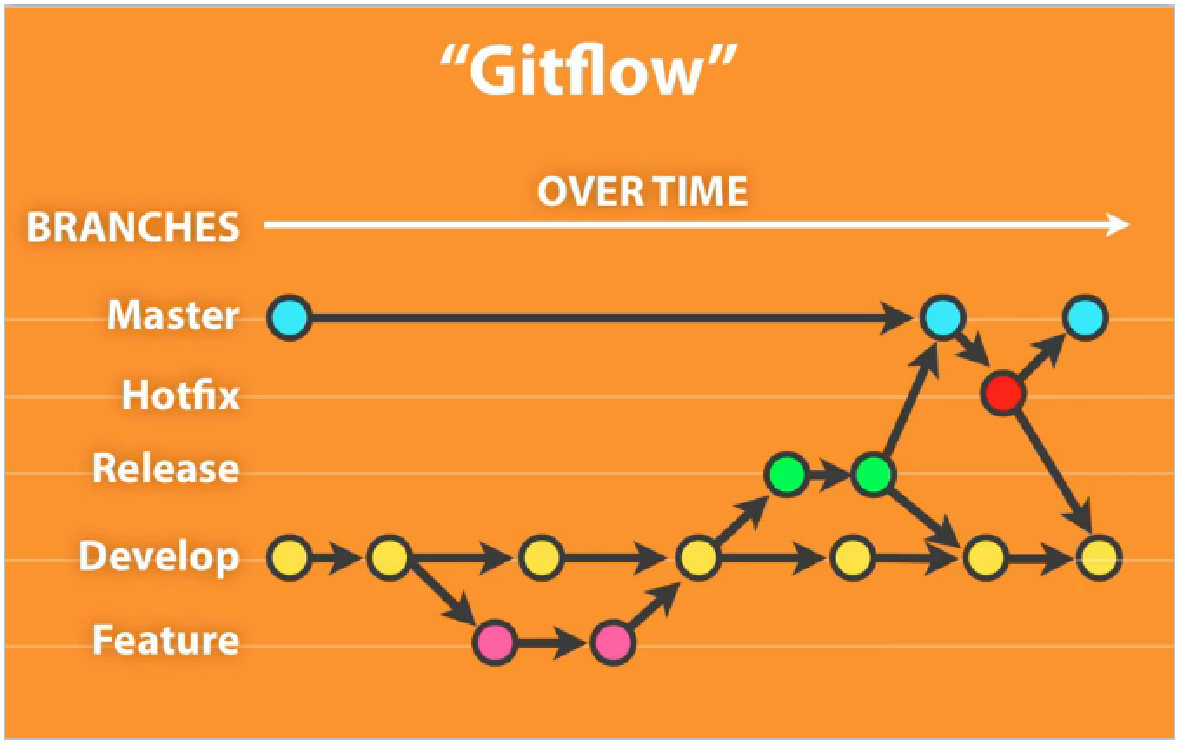
- ☑ Insert space after colon for base or interface in type declaration
- ☑ Insert space after comma
- ☐ Insert space after dot
- ☑ Insert space after semicolon in "for" statement
- ☑ Insert space before colon for base or interface in type declaration
- ☐ Insert space before comma
- ☐ Insert space before dot
- ☐ Insert space before semicolon in "for" statement

```
int Method()
{
    Console.WriteLine( "In Method" );
    return 3;
}
```

☑ Show all settings

Git Branching Strategy

Rock is using a branching model described Vincent Driessen now called Gitflow. You can read more about Gitflow and Vincent's corresponding Git extensions on the GitHub git-flow project. There is also a good 6 minute high-level overview video if Gitflow is new to you.



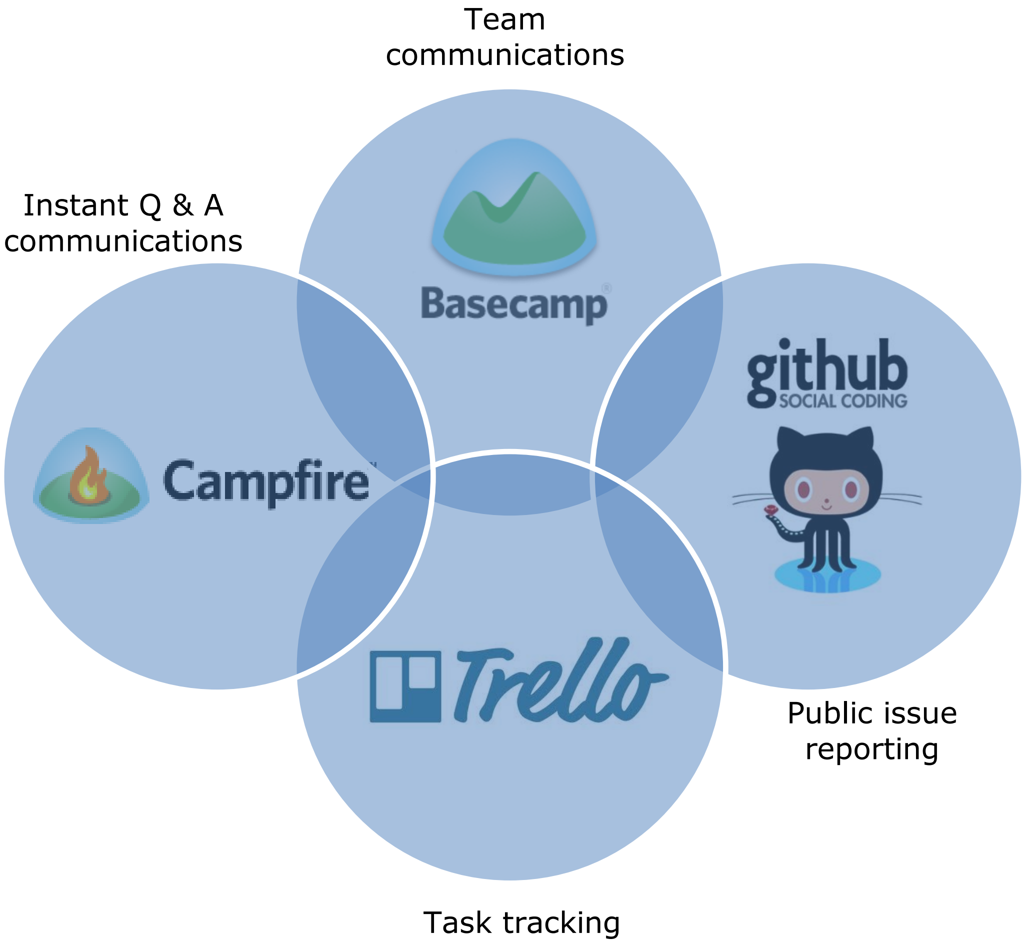
Once you have an understanding of Gitflow, read our Git Branching document which describes in detail exactly how we want you to operate.

Branch Naming Conventions

Type	Format	Example
Feature	feature-[your_initials]-[name-of-feature]	feature-drt-add-campus
Release	release-[version-number]	release-1.0
Hotfix	hotfix-[version-number]	hotfix-1.0.1
Custom	custom-[organization domain]	custom-ccvonline

Task Tracking & Communication

We use GitHub for tracking public issues, Trello for tracking project team tasks, and are currently using Campfire for immediate team member Q & A communications and Basecamp general purpose, large team communications as well as technology/feature discussions and witeboarding.



Required Reading

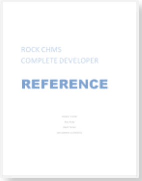
This Handbook

Ok, you’re almost done so don’t get too flustered about it.

Git Branching

As mentioned in the Git Branching Strategy section.

Developer Reference



ROCK CHMS
COMPLETE DEVELOPER

REFERENCE

Found in the **Documentation** folder, the Developer Reference is not only good for you, it is required reading. Seriously though – it is the best way to get ramped up developing proper in Rock ChMS.

The Framework API

This is found in your Documentation folder’s SandCastle\index.html

The REST API

This is found on your Rock environment under ~/REST/help

Developer Communication

Here are the best ways to communicate with the team and the Rock ChMS developer community.

Developer Lounge in Campfire
https://sparkdevnetwork.campfirenow.com/40198

IRC Channel #RockChMS
irc://chat.freenode.net/#RockChMS

Twitter - Follow us:
@RockChMS
@SparkDevNetwork
@RefreshCache

TBD - Developer forums on our community site

Github Wiki (use when you need to document something you create)
https://github.com/SparkDevNetwork/Rock-ChMS/wiki/Reference