|  |
| --- |
| СПбГУ, Математико-Механический факультет |
| Основы работы с системой контроля версий Git |
| Версия 1.0 |

|  |
| --- |
| Сартасов С.Ю.  9-13-2015 |

# История версий

|  |  |  |  |
| --- | --- | --- | --- |
| Версия | Дата | Автор | Описание |
| 1.0 | 13.09.2015 | Сартасов С.Ю. | Первое описание основных операци по работе с Git |

Содержание

[История версий 2](#_Toc429933806)

[Введение 4](#_Toc429933807)

[Создание локального репозитория 5](#_Toc429933808)

[Создание ветки 7](#_Toc429933809)

[Смена ветки 9](#_Toc429933810)

[Слияние веток 10](#_Toc429933811)

[Обновление данных с сервера 12](#_Toc429933812)

[Сохранение изменений на сервер 13](#_Toc429933813)

[Отмена изменений 17](#_Toc429933814)

# Введение

Git (<https://git-scm.com/>) – система контроля версий, разработанная Линусом Торвальдсом для управления разработкой ядра Linux. Это набор программ, который позволяет хранить исходные коды программ вместе с историей их изменения, а также заниматься совместной разработкой без необходимости передавать друг другу файлы с изменёнными исходными кодами. Любой серьёзный проект для достижения этих целей использует ту или иную систему контроля версий: Git, SVN, Visual SourceSafe и т.д.

Для использования Git необходимо установить следующие программы:

* Git for Windows – сама система контроля версий - <https://git-scm.com/download/win>
* (Опционально) TortoiseGit – графический клиент, встраивающийся в контекстное меню Windows - <https://tortoisegit.org/download/>

В Git все исходные коды и сопутствующие файлы хранятся в **репозитории** – выделенном хранилище на физическом носителе. Пользователи могут создавать на своих машинах копии этих репозиториев, добавлять туда свои исходные данные и отправлять эти изменения в удалённые репозитории (**commit** и **push**). Каждое такое изменение сопровождается порядковым номером – версией. В ряде случаев для удобства работы можно скопировать исходные коды некоторой версии – создать **ветку (branch)**. Достоинство веток состоит в том, что их изменения добавляются и отслеживаются независимо друг от друга, но всегда есть возможность посмотреть, на какой версии произошло ветвление репозитория на две и более веток. Если же есть необходимость добавить в одну ветку данные из другой (например, если изменения во второй ветке облегчают разработку в первой), для этого используется операция **слияния (merge)**.

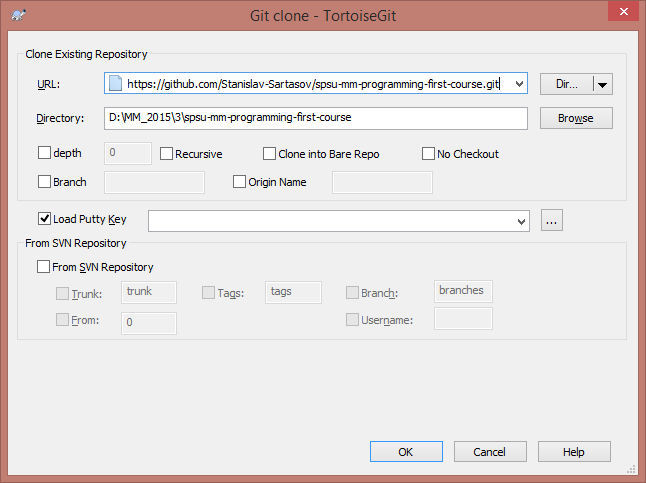
В данном документе описываются основные операции, необходимые при работе с Git.

# Создание локального репозитория

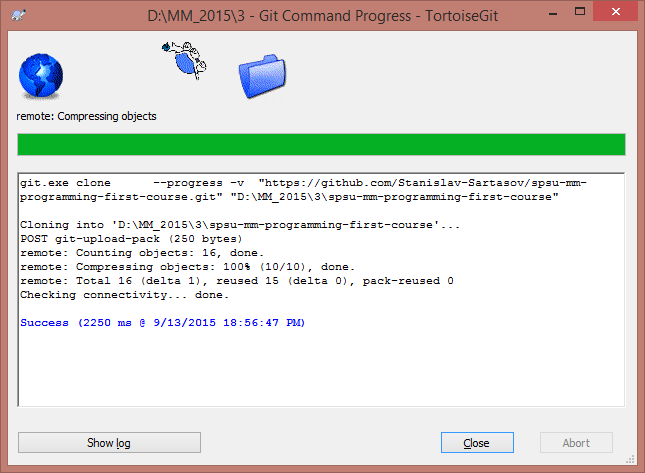
Для начала работы в Git необходимо настроить папку как локальный репозиторий и загрузить в него текущую версию удалённого репозитория. Это делается с помощью команды Git Clone:



В открывшемся окне необходимо указать адрес удалённого репозитория и директорию, в которой будет создан локальный репозиторий. Остальные параметры можно оставить без изменений:

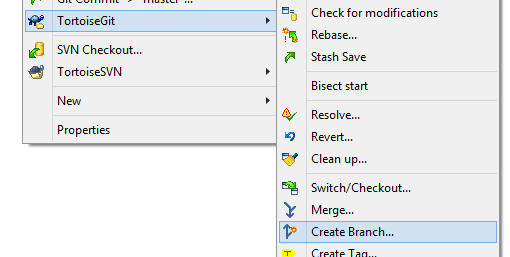


В результате выполнения этой команды на диск будет загружена ветка master удалённого репозитория:



# Создание ветки

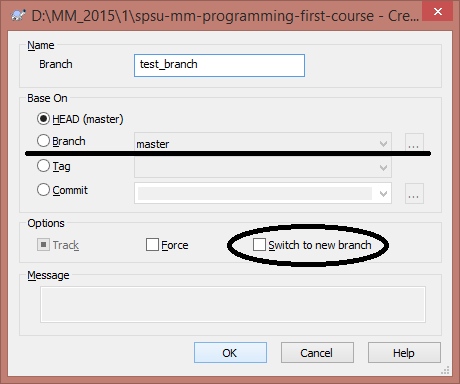
Чтобы создать свою ветку, необходимо выбрать команду Create new branch:



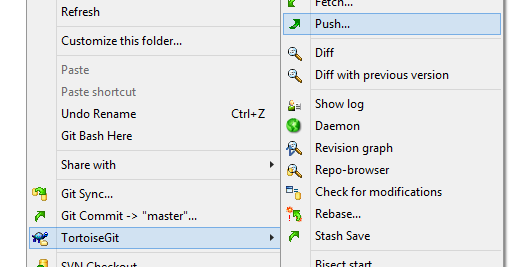
В открывшемся окне необходимо указать имя новой ветки, а также исходную ветку и при необходимости версию. Наиболее часто используются первые два варианта:

1. Создать ветку на основе последней версии текущей ветки локального репозитория.
2. Создать версию на основе последней версии выбранной ветки локального или удалённого репозитория.

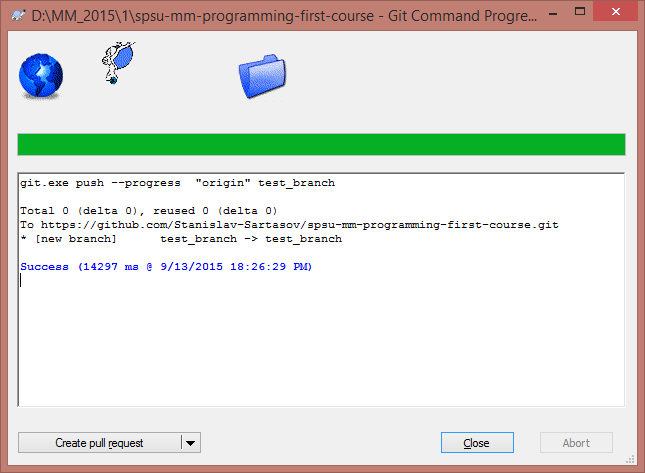
Выбор «Switch to new branch» позволяет сразу же после создания сделать новую ветку текущей веткой локального репозитория. Если галочка не поставлена, то после создания ветки текущая ветка локального репозитория останется прежней.



Новая ветка создана в локальном репозитории, поэтому для того, чтобы сделать её видимой другим пользователям, требуется выполнить команду Push:

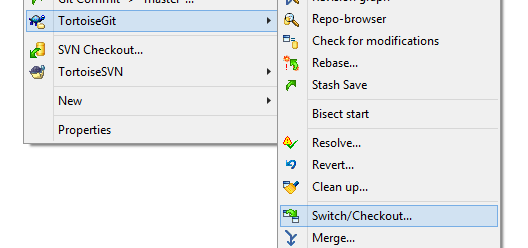


В окне с результатом будет показано, что ветка была добавлена в удалённый репозиторий:

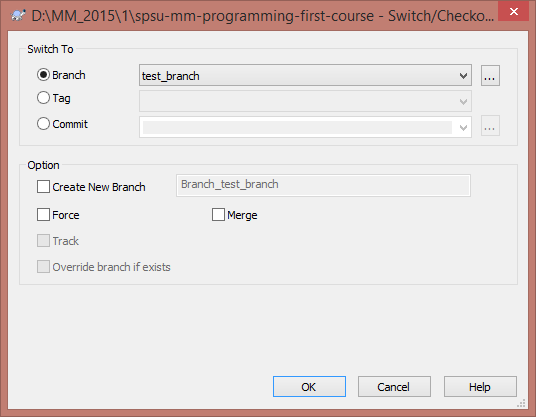


# Смена ветки

Для того, чтобы сменить ветку, с которой работает локальный репозиторий, необходимо выбрать команду Switch/Checkout:



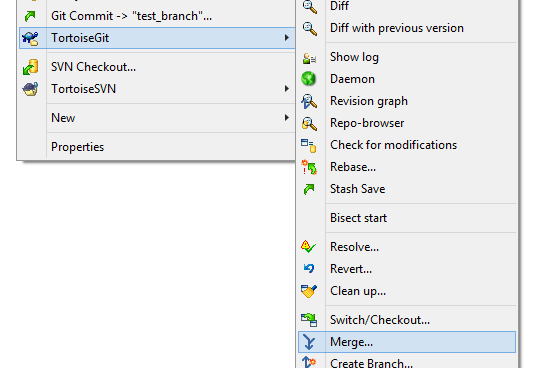
В открывшемся диалоговом окне необходимо выбрать, на какую ветку удалённого репозитория необходимо перейти (Switch To):



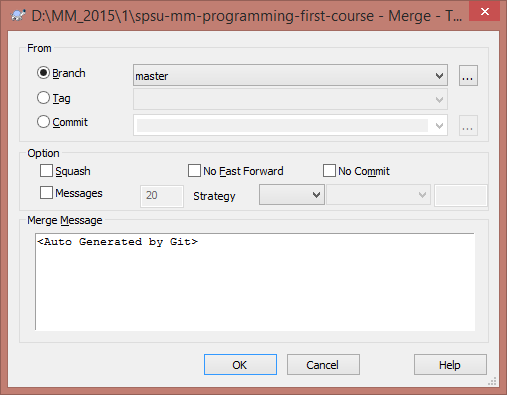
Ветки, название которых начинается с Remote/Origin/ - это ветки удалённого репозитория. Остальные – ветки локального репозитория.

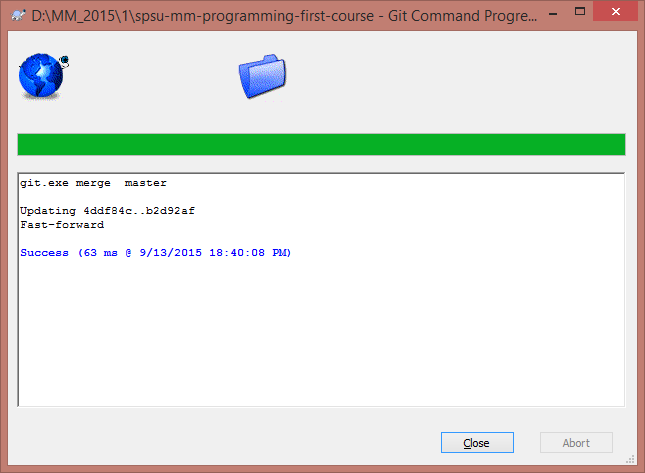
# Слияние веток

Для слияния веток выбирается пункт Merge:



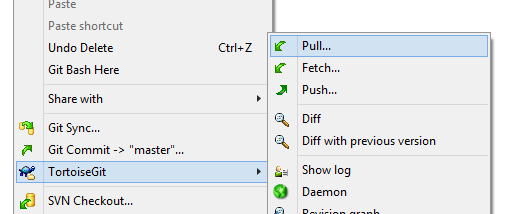
В открывшемся окне указывается, из какой ветки локального или удалённого репозитория получать данные для слияния их с данными текущей ветки локального репозитория.

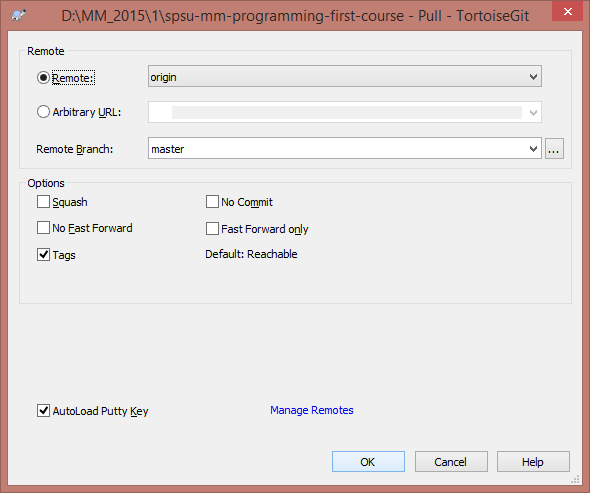




# Обновление данных с сервера

Для обновления текущей ветки локального репозитория данными из соответствующей ветки удалённого репозитория используется команда Pull. Такая необходимость регулярно возникает при совместной работе нескольких человек над одной и той же веткой:



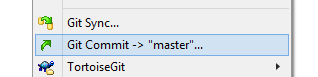


Иногда обновление даных оканчивается неудачей. В этом случае вам будет показано, какие файлы препятствуют обновлению (например, в них содержатся не подлежащие автоматическому разрешению локальные изменения, не сохранённые в локальном или удалённом репозитории, или произошёл конфликт содержимого между вашей версией и версией удалённого репозитория). В этом случае необходимо либо отменить изменения в этих файлах, либо вручную их отредактировать.

# Сохранение изменений на сервер

Чтобы изменённые вами файлы стали новой версией в репозитории, необходимо их сперва добавить в локальный репозиторий, а потом в удалённый.

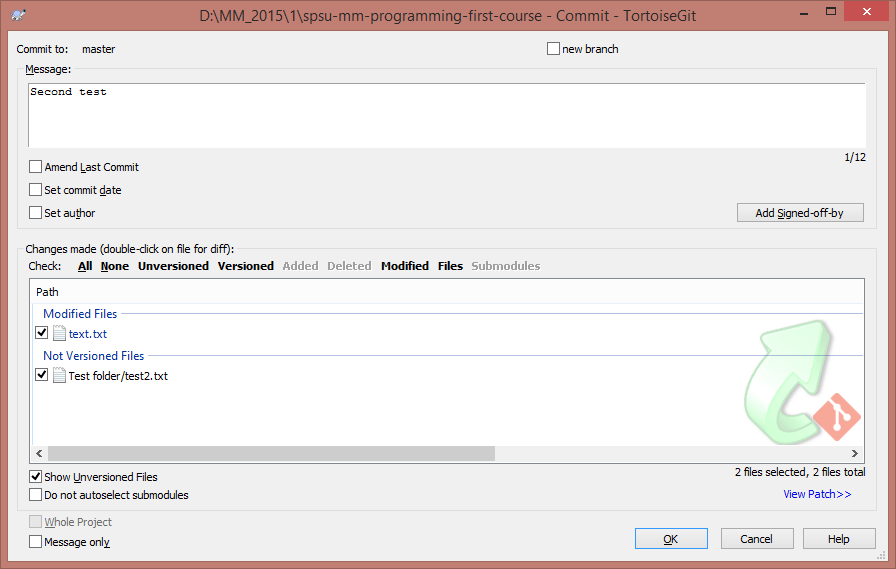
Для сохранения изменений в локальный репозиторий (жарг. commit, коммит), необходимо воспользоваться командой Commit:



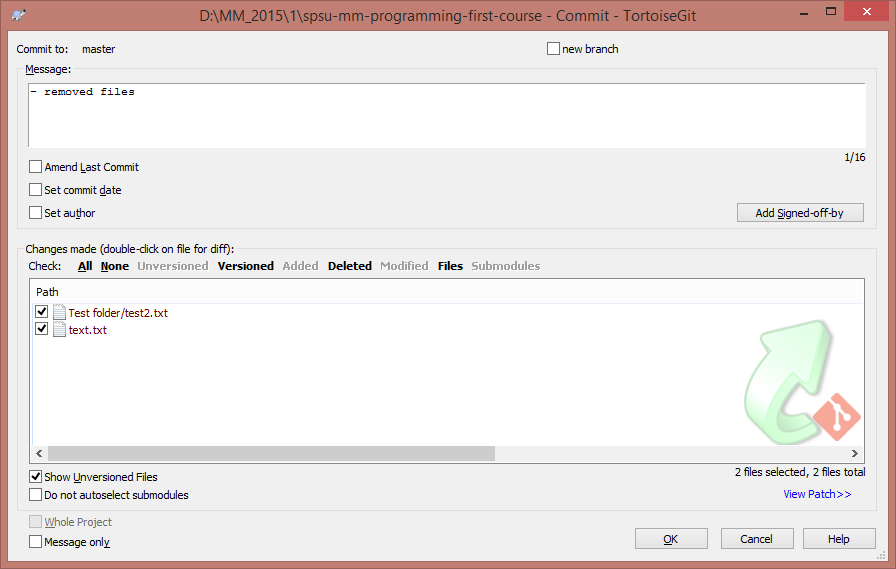
В открывшемся окне необходимо отметить те файлы, которые вы хотите сохранить в репозитории, при этом добавление новых файлов не отличается от сохранения изменений в старых.

**Важно!** Сохранять все доступные файлы в репозиторий, включая, к примеру, файлы с пользовательскими настройками проекта, – это осуждаемая в реальных проектах практика.

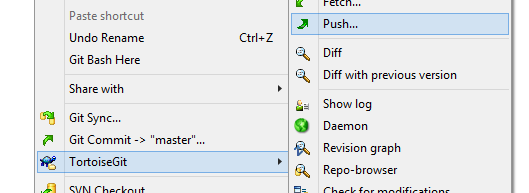
**Важно!** Каждый коммит должен сопровождаться содержательным комментарием. С примерами бессодержательных комментариев можно ознакомиться на сайте <http://whatthecommit.com/>.



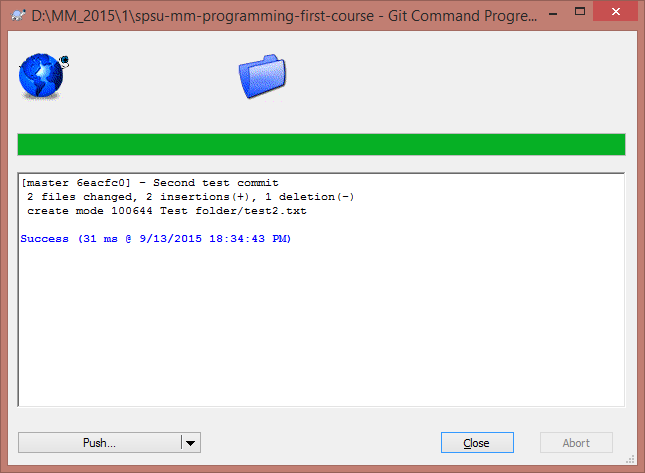
Для удаления файлов из репозитория нужно удалить их из файловой системы, а потом сохранить это изменение командой Commit:



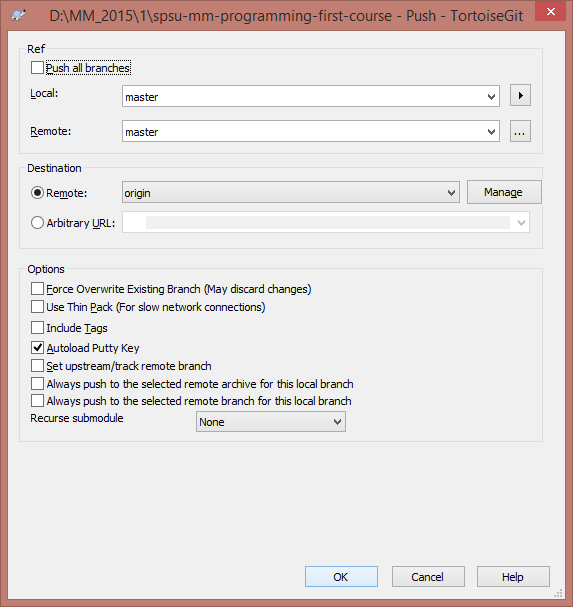
После сохранения изменений в локальном репозитории, их необходимо поместить в удалённый репозиторий командой Push:

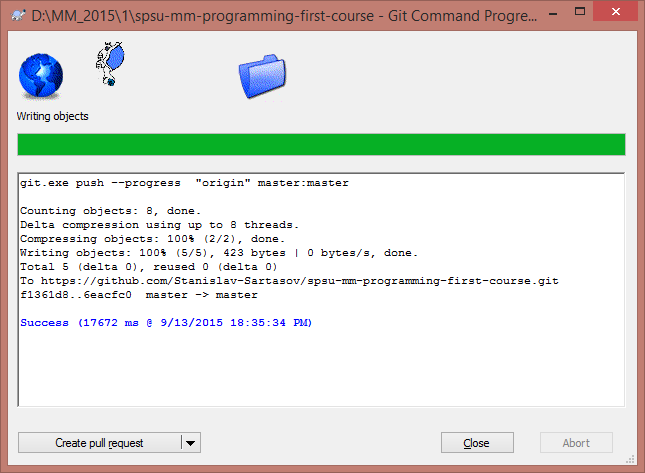


Также эту команду можно вызвать после удачного коммита:



Чаще всего нет необходимости изменять какие бы то ни было параметры в появившемся окне:

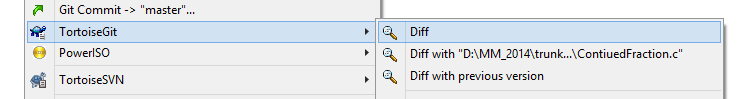




Наиболее частой причиной, по которой изменения не удаётся сохранить на удалённом сервере, является наличие новой версии, добавленной кем-то другим. В этом случае необходимо обновить версию из удалённого репозитория (и разрешить конфликты, если таковые возникнут).

# Отмена изменений

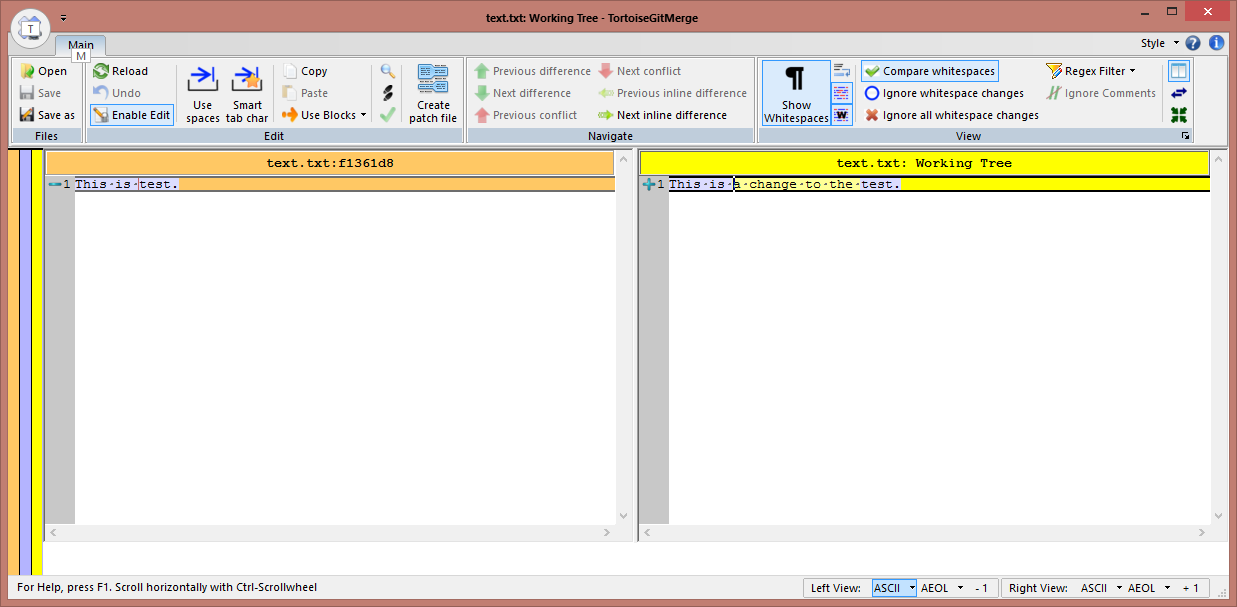
Иногда бывает полезно посмотреть, какие изменения были сделаны в файле, но ещё не были сохранены в репозитории. Для этого используется команда diff:



У этой команды есть три варианта:

* Diff – сравнение с текущей версией файла в репозитории;
* Diff with previous version - сравнение с предыдущей версией файла в репозитории;
* Diff with %FILENAME% - сравнение между собой двух файлов.

В появившемся окне справа отображается выбранный файл, а слева – версия из репозитория или файл, с которым сравнивают выбранный. При этом оранжевым помечаются удалённые области текста, жёлтым – добавленные, красным – области конфликтов. Можно выбирать области текста и с помощью контекстного меню заменять части файла справа на части файла слева.



Сохранение файла после изменений приводит только к сохранению его на жёстком диске. Добавление новой его версии в репозиторий делается как обычно с помощью команд Commit и Push.

Перед коммитом бывает полезно проверять таким образом все изменения, которые сохраняются в репозитории.

Если же возникает необходимость откатить все текущие изменения в файле к его последней версии в локальном репозитории, это можно сделать командой Revert:

