**Начинаем работать с git — пошаговая инструкция**

16.10.2018Татьяна Георгиевская

Первые шаги в освоении мира компьютерных технологий пройдены, ты знаешь, зачем сюда пришёл, к чему стремишься, и как этого добиться, упорно учишься, переходя от курса к курсу. За спиной первые написанные программы и отловленные баги, а впереди маячат светлые перспективы коммерческой разработки программного обеспечения.

Наверное, пора узнать…

**Содержание**

* [Секреты командной разработки](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#i)
* [Шаг 1. Выбираем git-хостинг](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#_1_git)
* [Шаг 2. Регистрация](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#_2)
* [Шаг 3. Создание репозитория](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#_3)
* [Шаг 4. Работа с репозиторием](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#_4)
* [Шаг 5. Выбираем Гит-клиент](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#_5)
  + [SmartGit](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#SmartGit)
  + [GitHub Desktop](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#GitHub_Desktop)
  + [GitKraken](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#GitKraken)
  + [SourceTree](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#SourceTree)
* [Шаг 6. Работа со SmartGit](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#_6__SmartGit)
* [Основные операции для работы с git](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#___git)
  + [Clone](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#Clone)
  + [Commit](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#Commit)
  + [Push](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#Push)
  + [Pull](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#Pull)
  + [Перенос информации из сторонних репозиториев на Гитхаб](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#i-2)
* [Заключение](https://progtips.ru/instrumenty-programmista/nachinaem-rabotat-s-git-poshagovaya-instrukciya.html#i-3)

**Секреты командной разработки**

Разработка – это почти всегда командная игра. Пора учиться работать в команде.  
Даже если пока что в твоей команде только монитор, системник (или старенький ноутбук) и острое желание стать программистом, всё равно пора учиться.

Программисту проще стать своим среди своих, ведь за него говорят не дежурные улыбки, перекуры и чаепития с печеньками, а чёткие строки кода, элегантные функции и безупречная работа готовых приложений.

Чтобы эффективно работать в команде, мало знать синтаксис языка, ключевые библиотеки и уметь обращаться с базами данных. Необходимо уметь работать в удобной для команды системе контроля версий.

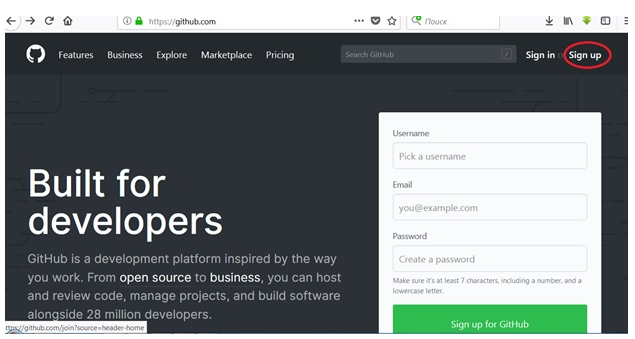
О системах контроля версий их преимуществах и недостатках можно почитать [здесь](https://progtips.ru/instrumenty-programmista/sistemy-kontrolya-versij-osnova-komandnoj-razrabotki.html).  
В этой статье мы перейдём от теории к практике и расскажем, как работать с git’ом.

**Шаг 1. Выбираем git-хостинг**

Git-хостинг на разных условиях предлагают десятки компаний.  
Самые известные из них: [Github](https://github.com/), [Sourceforge](https://sourceforge.net/), [Google Code](https://code.google.com/),  [GitLab](https://gitlab.com/), [Codebase](https://www.codebasehq.com/). Выбери удобный интерфейс и регистрируйся на понравившемся хостинге.  
В этой статье мы рассмотрим работу с git-хостингом на примере [Github](https://github.com/)’а.

**Шаг 2. Регистрация**

Процедура регистрации на Гитхабе  простая и интуитивно понятная даже для тех, чей уровень английского далёк от  Upper Intermediate.



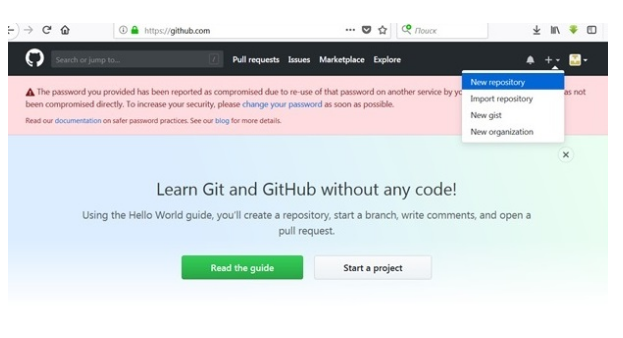
Логин, пароль, почта –> подтверждение, и  связь с мировым сообществом программистов налажена.

**Шаг 3. Создание репозитория**

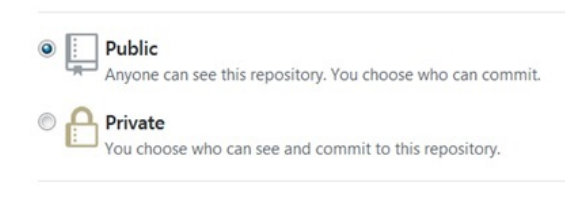
Ты можешь создать любое количество репозиториев, у каждого из которых будет issue tracking, wiki, условия для проведения code review и многое другое.  
Политика разработчиков Github предполагает бесплатное использование хостинга для всех open-source проектов.

Чтобы создать новый репозиторий нажмём кнопку **+**в верхней части экрана и выберем **New repository**

Создание репозитория на Гитхабе

Многие разработчики рано или поздно сталкиваются с необходимостью создания приватного репозитория, код из которого доступен только их команде. Для этих случаев  на Github’е есть определённый тарифный план.

Но пока острой необходимости в создании приватного репозитория у нас нет, создадим обычный.



Жмём волшебную кнопку **Create** внизу экрана, и репозиторий готов.

**Шаг 4. Работа с репозиторием**

Работа с репозиторием может вестись из командной строки, напрямую из среды разработки или из графического интерфейса (git — клиент приложения).

Работа с графическим интерфейсом позволяет лучше понимать процессы, происходящие в локальном и удалённом репозитории. Поэтому я рекомендую начать работу с git с использованием графического интерфейса.

**Шаг 5. Выбираем Гит-клиент**

Потом, когда суть процессов изменения и обновления (восстановления) информации в репозитории станет для Вас очевидна, можно работать и через командную строку. В этом принципе работы есть немало своих преимуществ. Например, все новые опции Гитхаба реализуются сначала для использования в командной строке, и только потом адаптируются под графические интерфейсы.

Но вернёмся к git-клиентам.

Самыми популярными гит- клиентами на данный момент являются:

**SmartGit**

Удобное  приложение гармонично сочетает все необходимые функции и доступную интуитивно понятную систему управления. SmartGit – один из самых удобных графических интерфейсов для профессиональной разработки. Некоммерческая разработка и разработка open-sourse проектов не требуют платной лицензии.

**GitHub Desktop**

«Родной» графический интерфейс Гитхаба. GitHub Desktop работает под Windows и Mac и практически полностью копирует функционал основного сайта. Работает под той же учётной записью.  
Правда, [не всегда оперативно справляется с большими программами.](https://twitter.com/steipete/status/631491465240145920/photo/1)

Зато отлично подходит для начала работы с git.

**GitKraken**

Поддерживает GitHub, Bitbucket и Gitlab.  
Кракен очень любят программисты – фрилансеры, которым периодически приходится менять команды, а значит, и условия командной разработки. Возможность работы с разными git-хостингами через привычное приложение со знакомым интерфейсом в таких случаях играет важную роль.

**SourceTree**

SourceTree позволяет работать с Bitbucket и GitHub. В приложении довольно простой интерфейс, подходящий, как для опытных программистов, так и для новичков.

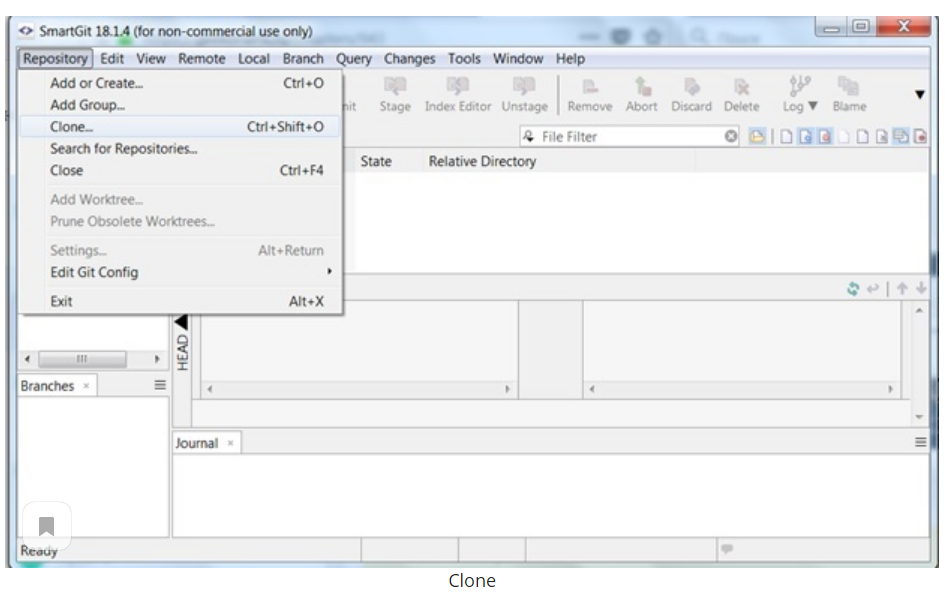
**Шаг 6. Работа со SmartGit**

В этой статье мы рассмотрим работу с SmartGit.Скачать SmartGit можно, например, [отсюда](https://www.syntevo.com/smartgit/):  
Устанавливаем и запускаем.

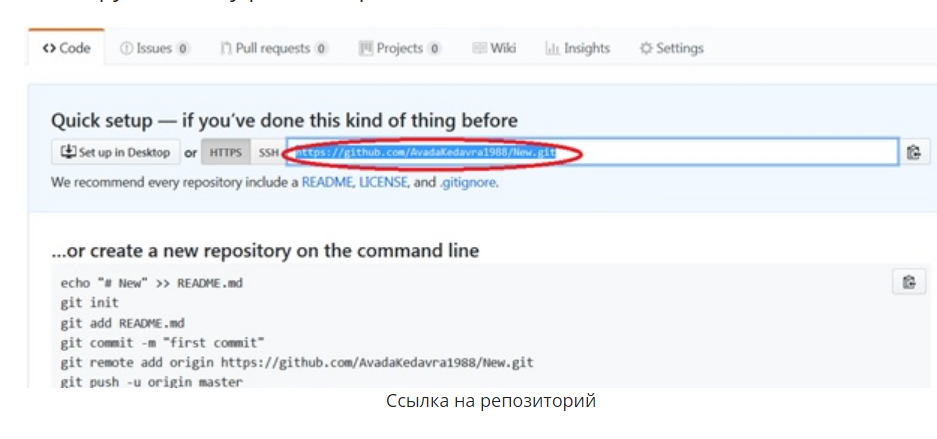
**Основные операции для работы с git**

**Clone**

Первое, чему стоит научиться – это снимать копию проекта из удалённого репозитория в локальный.  
Делается это довольно просто:



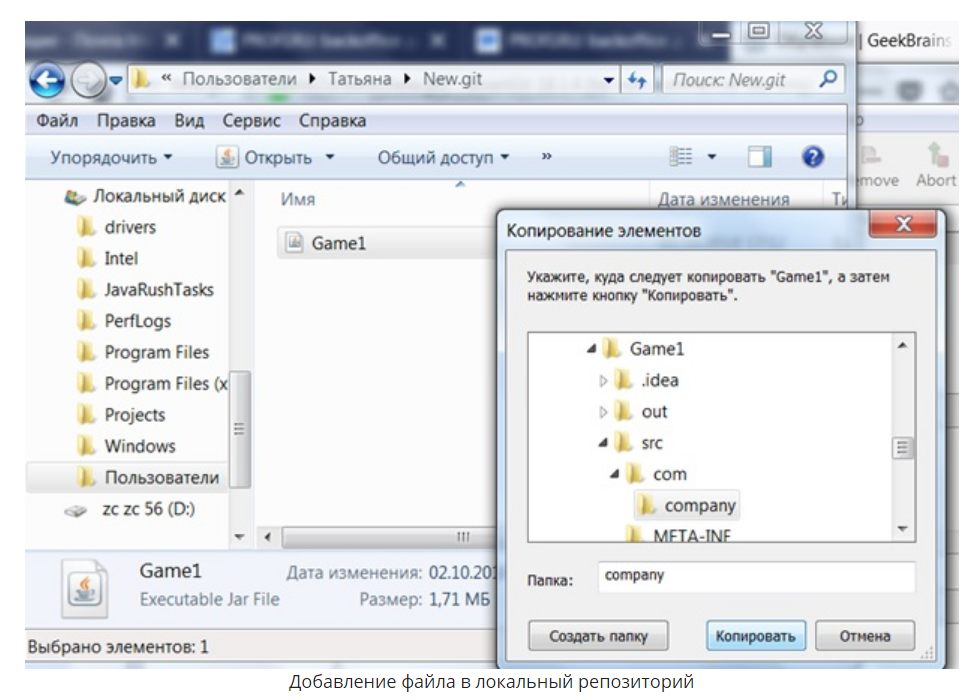
Затем копируем ссылку репозитория, созданного на Гитхабе (шаг 2)

Вставляем адрес удалённого репозитория в нужную ячейку в открывшемся окне, выбираем расположение нового локального репозитория у нас на компьютере, и получаем готовый локальный репозиторий.

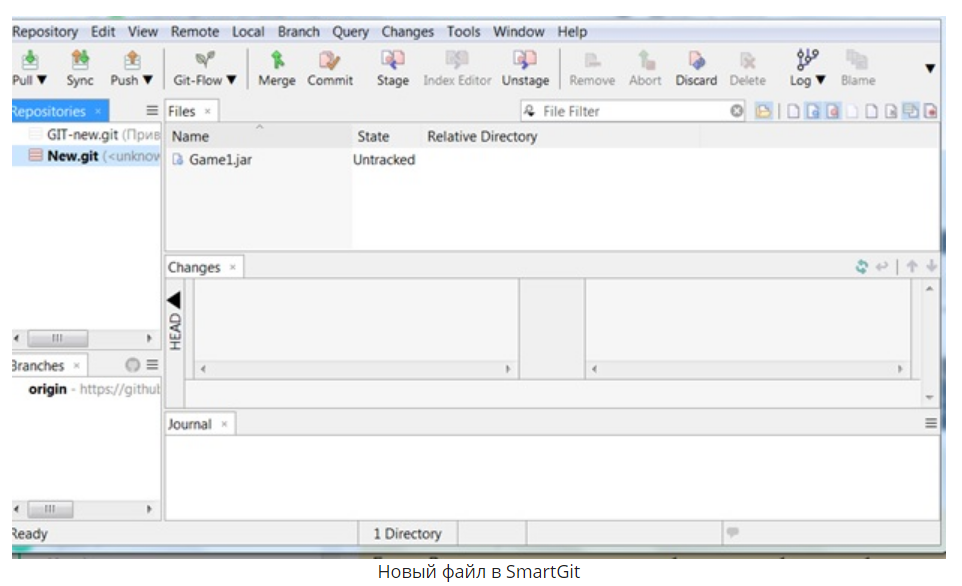
К слову, аналогичным образом можно клонировать чужой открытый репозиторий и поближе познакомиться с чужим кодом.

**Commit**

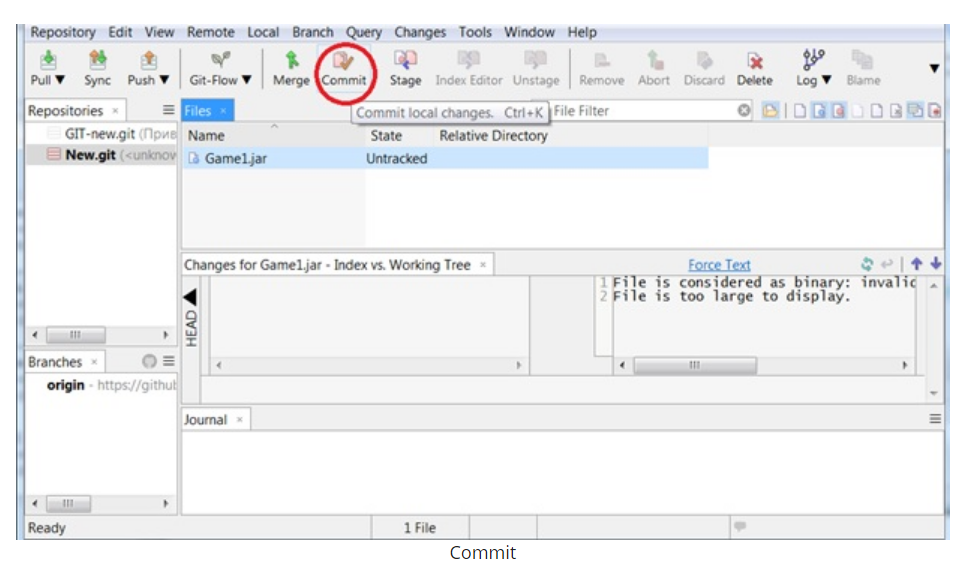
Репозиторий готов – пора приступать к работе.  
Написанный код мы помещаем в локальный репозиторий  — папку .git (путь к которой мы указали в операции clone).



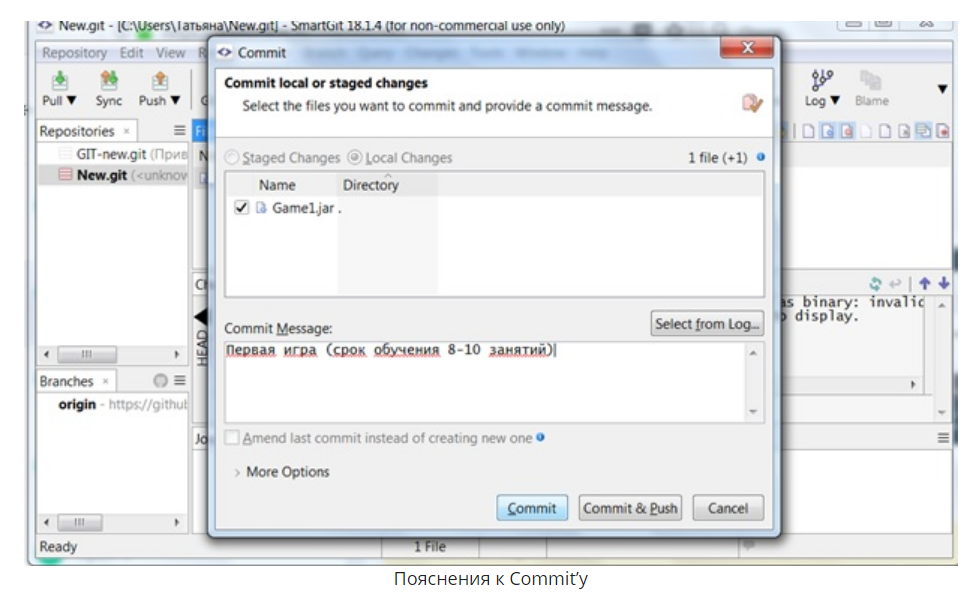
Если всё прошло успешно, в окошке SmartGit’а появится скопированный файл.

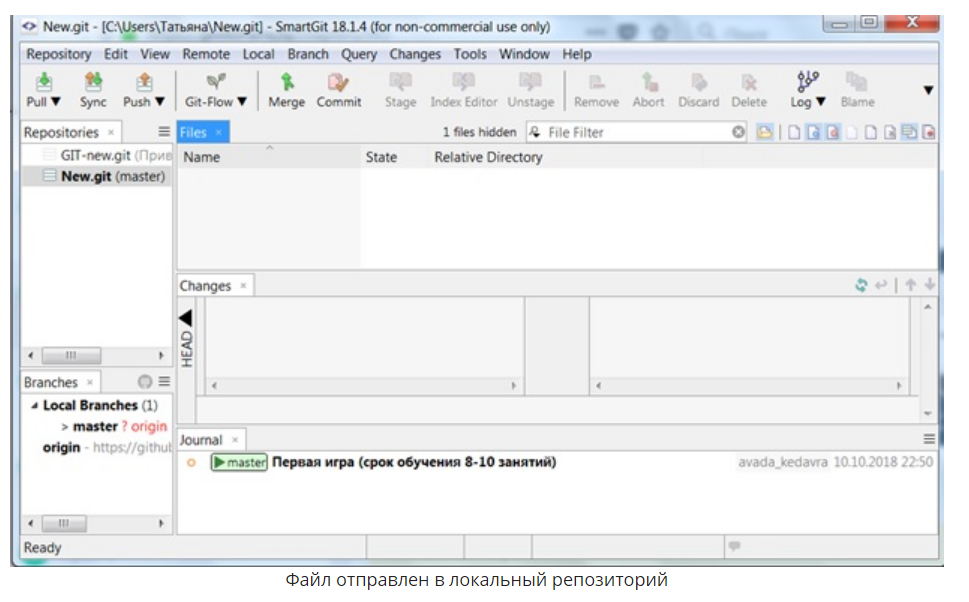


Для того чтобы зафиксировать изменения в локальном репозитории, нажимаем кнопку **Commit.**

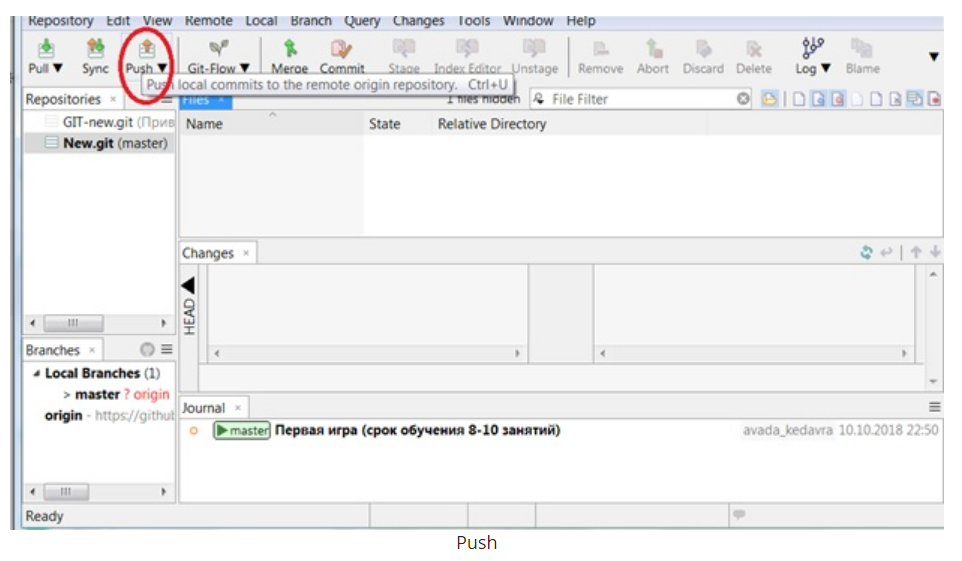


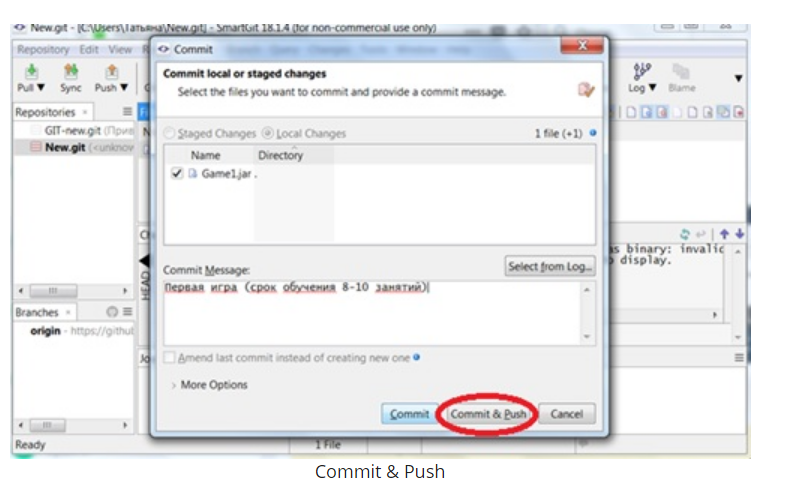
В открывшемся окне пишем пояснительный комментарий к сохраняемому файлу и снова нажимаем кнопку **Commit**

Файл сохранён, а изменения внесены в журнал.

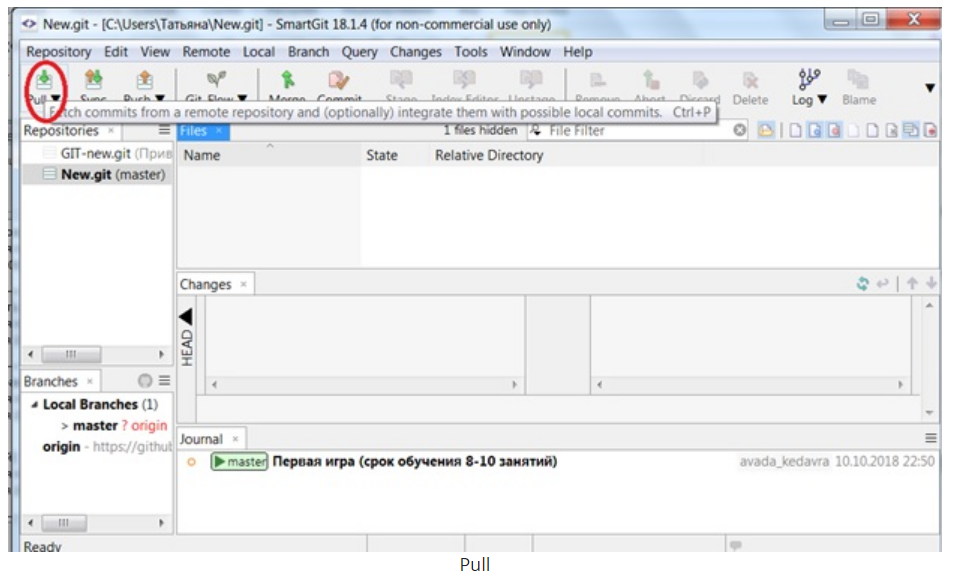
 **Push**

Теперь заглянем на [Github.com](https://github.com/) в наш удалённый репозиторий. Там до сих пор нет ни одного файла. Нужно срочно менять ситуацию.Чтобы перенести изменения, внесённые в локальный репозиторий, в удалённый репозиторий, необходимо нажать кнопку **Push.**

К слову, отправить изменения в удалённый репозиторий, нам предлагают ещё в точке **Commit**’а

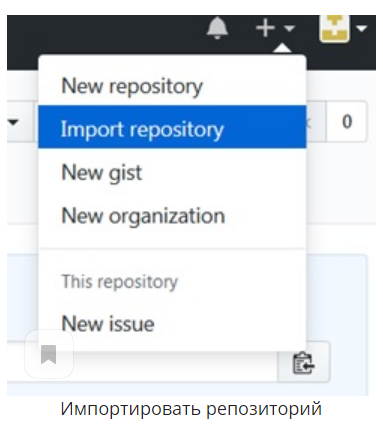
 **Pull**

Возникает резонный вопрос: как получат изменения остальные участники разработки, если они клонировали проект в самом начале?  
Для этого существует команда **Pull**, передающая в локальный репозиторий все изменения, происходящие в удалённом.

К слову, для командной разработки на Гитхабе есть ещё несколько важных опций.

**Перенос информации из сторонних репозиториев на Гитхаб**

Когда нужно собрать разрозненные кусочки кода в один проект, используйте кнопку **Import repository** и работайте с файлами в удобном репозитории Гитхаба.



Кнопка **New gist** на этой панели предназначена для мгновенного обмена информацией.

А кнопка **New organization** открывает массу возможностей для командной разработки.

**Заключение**

О git’е можно писать ещё долго, подробно рассматривая возможные конфликты, создание и  слияние деревьев, работу с ветками, но для начала эффективной работы достаточно знания основных команд и острого желания стать программистом.

Благодаря своей политике (поддержка open-sourse проектов) Github предоставляет удивительную возможность детально рассматривать программы, написанные как новичками, так и признанными гениями – программистами.

Искренне советую посмотреть, как пишут программный код профессионалы. Возможно, однажды отточенная профессиональная стилистика кода, вошедшая в проекты после знакомства с Гитхабом, поможет найти в бурлящем море вакансий работу твоей мечты.