

Зачем сдавать ПЗ через Git?

В современном мире практически ни одна командная разработка ПО не обходится без систем контроля версий. Git среди них – одна из самых популярных. Чем раньше начинающий разработчик начнет осваивать Git, тем лучше.

С помощью этой инструкции вы научитесь:

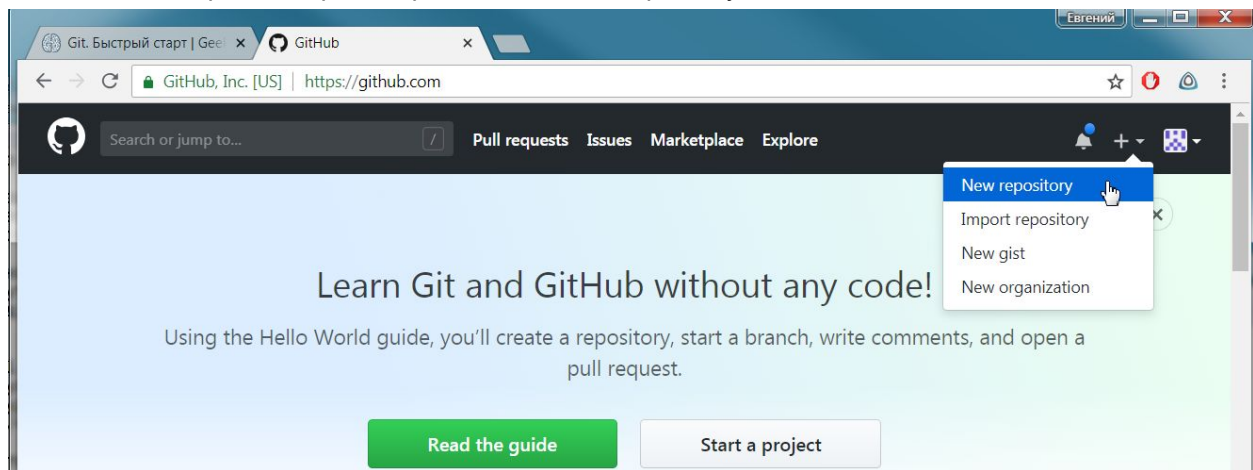
- создавать Git-репозиторий;
- использовать клиент SmartGit;
- выполнять операции Clone, Commit, Push, Pull;
- создавать pull-request-ы.

Инструкция очень сжатая. Более подробную информацию о Git-е вы можете получить, пройдя видеокурс [Git. Быстрый старт](#).

Создание Git-репозитория

Для начала зарегистрируемся на сайте [github](#).

Затем создайте репозиторий: Стрелочка -> New Repository



Укажите название репозитория и поставьте галочку на «Initialize this repository Readme». В качестве названия репозитория укажите название курса, для которого вы собираетесь делать практическое задание.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner



Repository name

/ python

Great repository names are short and memorable. Need inspiration? How about **psychic-couscous**.

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▼

Add a license: GNU General Public License v3.0 ▼



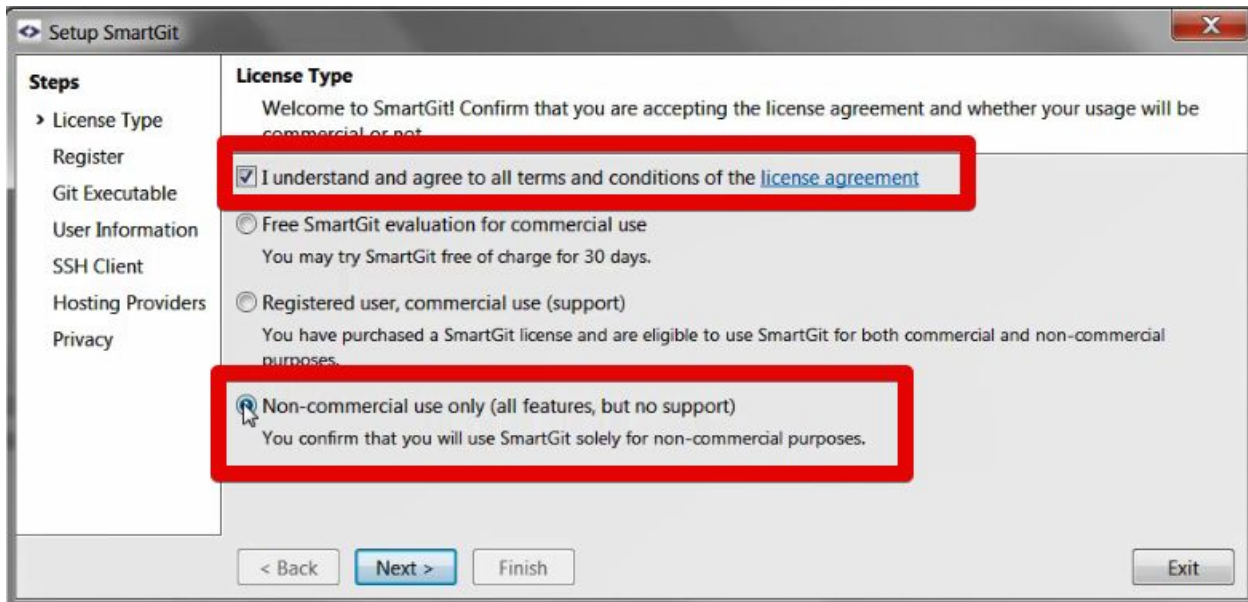
Create repository

Репозиторий готов!

Установка Git-клиента SmartGit

Для работы с Git-ом вы можете использовать любой Git-клиент, который вам нравится. В этой инструкции мы рассмотрим работу с программой SmartGit. Скачайте приложение [SmartGit](#) для своей операционной системы и запустите установку.

В процессе установки согласитесь с лицензионным соглашением и выберите опцию «Non-commercial use only» – в этом случае программа SmartGit будет для вас бесплатной.



Укажите свой UserName и E-mail, затем выберите опцию «Use SmartGit as SSH client».

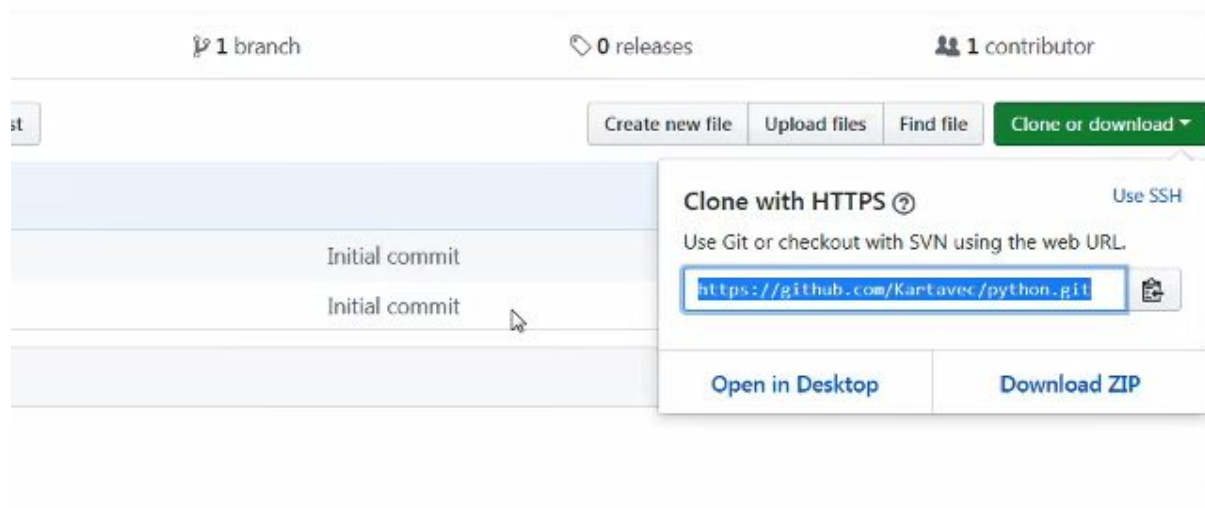


Нажимаем Next, Next, Finish.

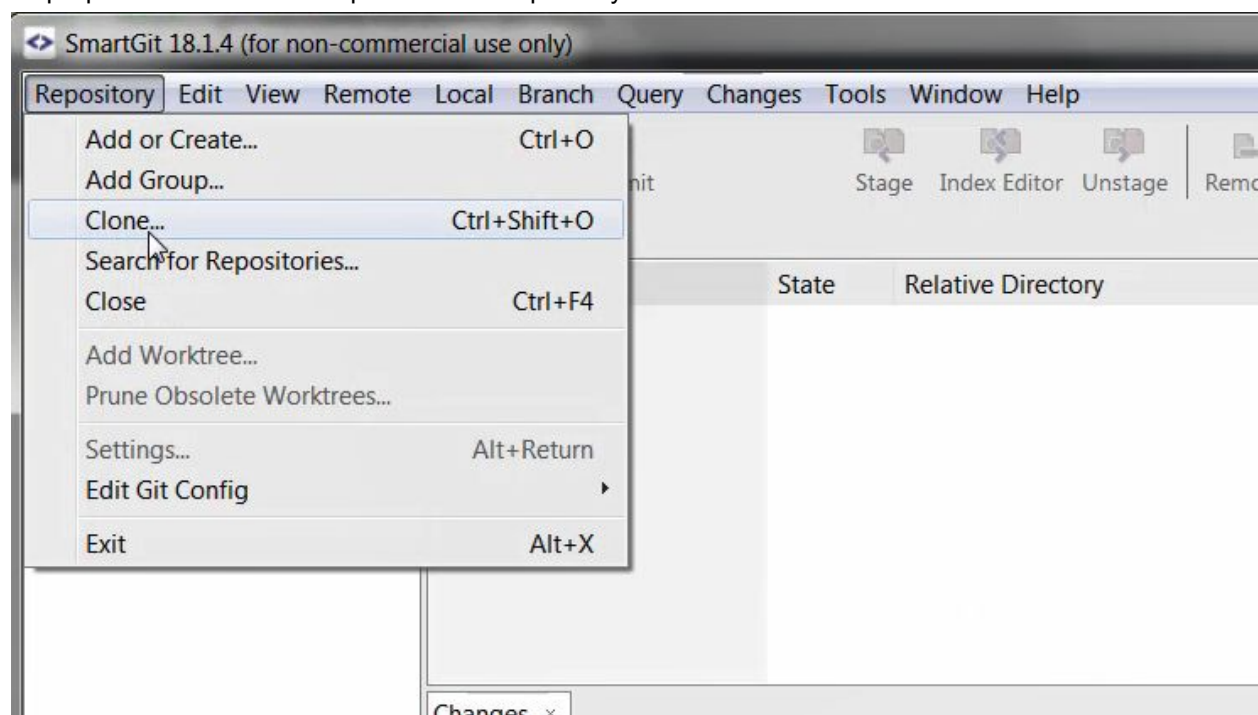
Клонирование репозитория

Заходим на страницу репозитория в GitHub (в моем случае это <https://github.com/Kartavec/python>).

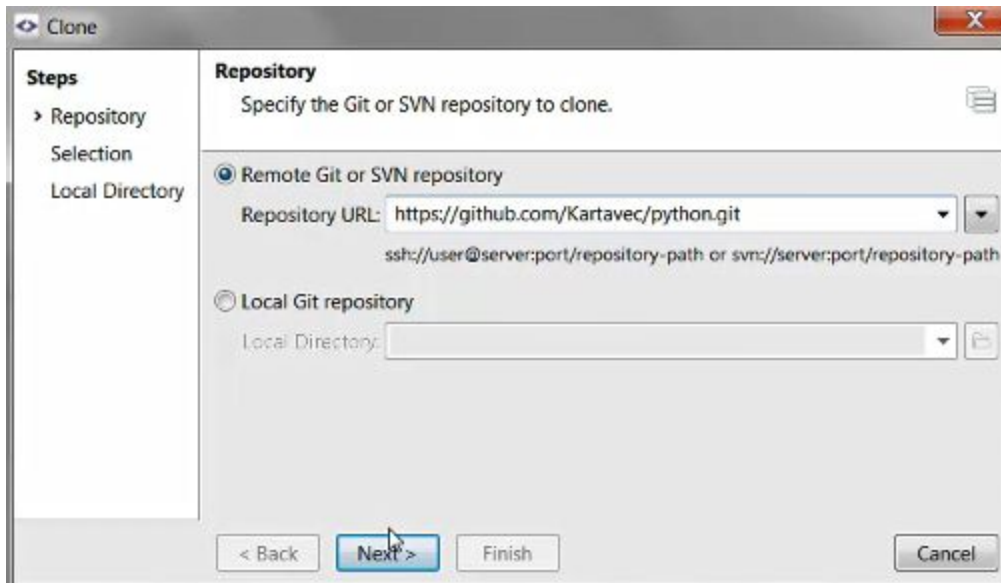
Нажимаем Clone or download, копируем ссылку в буфер обмена.



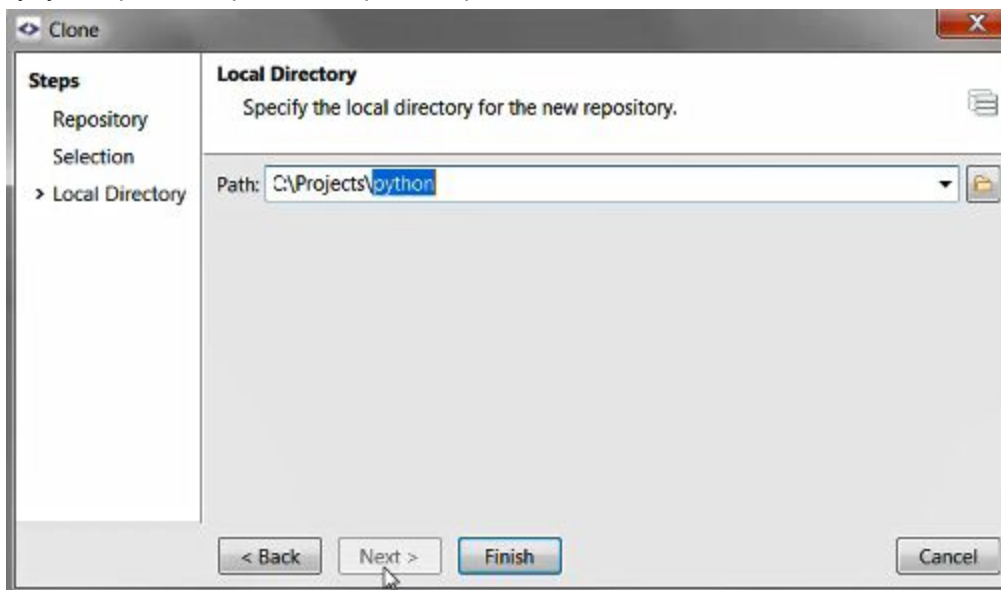
В программе SmartGit выбираем меню Repository -> Clone.



Вставляем ссылку на ваш репозиторий:



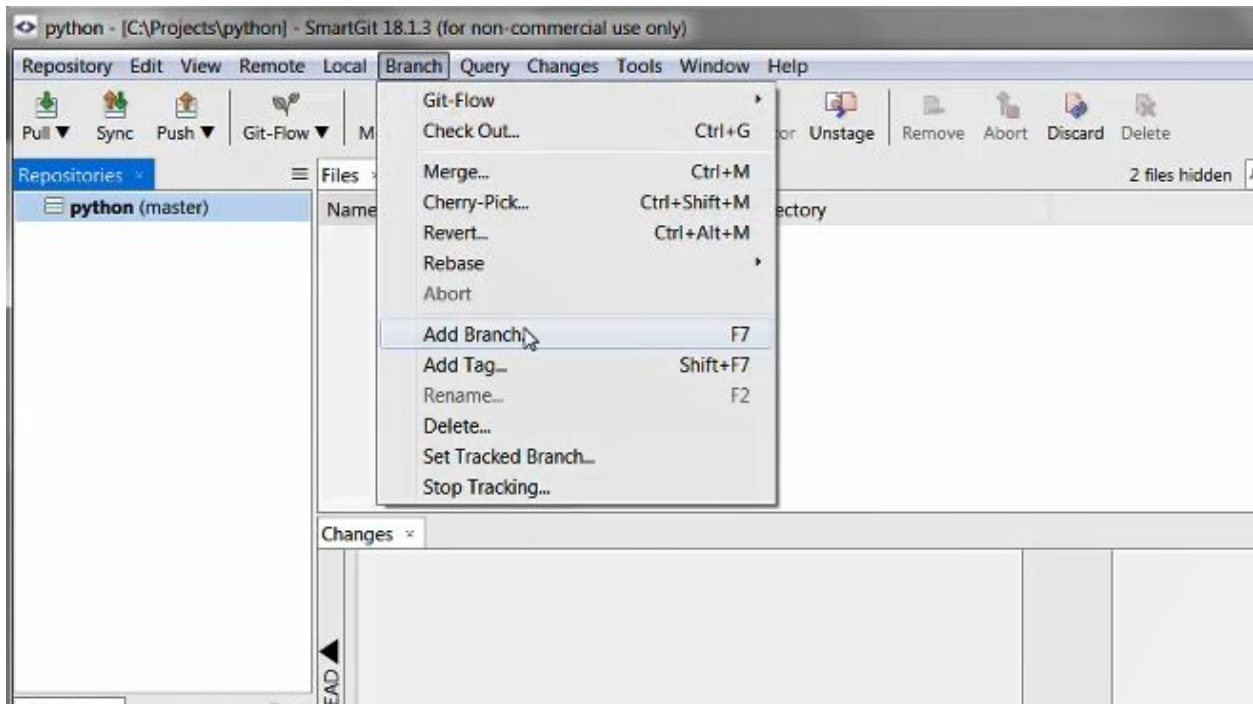
Нажимаем Next, Next. Выбираем путь к какой-либо пустой папке, в которой будет храниться будущий репозиторий и все файлы практических заданий.



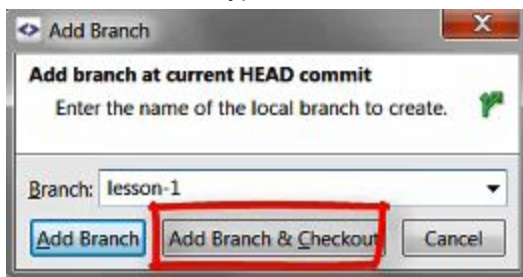
Репозиторий клонирован на локальный компьютер!

Делаем практическое задание к 1-му уроку

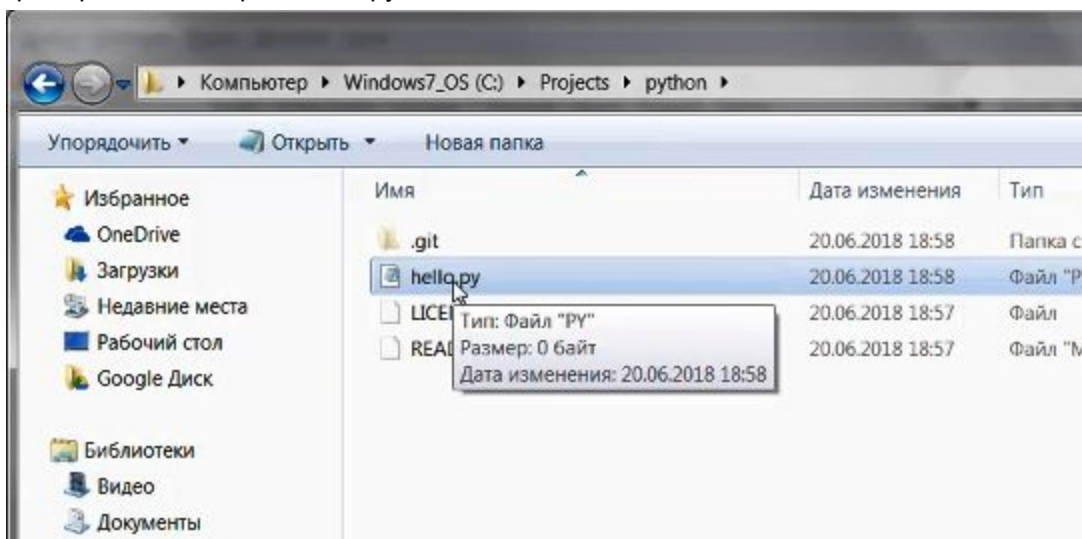
Для каждого урока мы будем создавать отдельную ветку (подробнее о ветках смотрите в курсе [Git. Быстрый старт](#)). В приложении SmartGit выбираем меню Branch -> Add Branch



Вводим название урока и нажимаем «Add Branch & Checkout»:

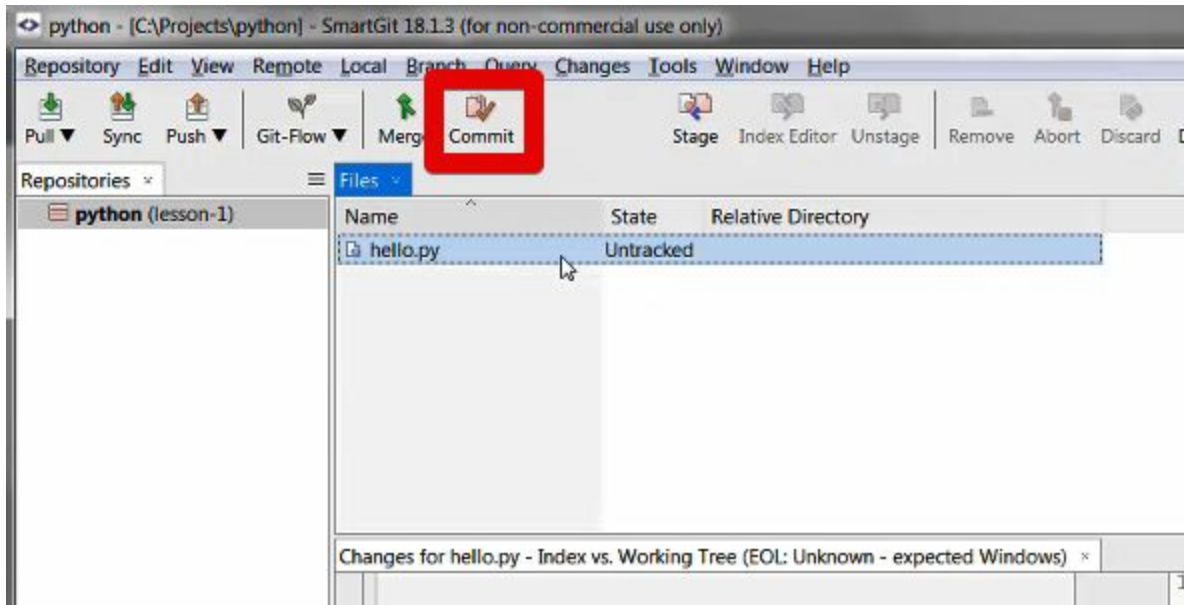


Переходим в папку, где хранится репозиторий, и выполняем практическое задание. В качестве примера я создал файл hello.py:

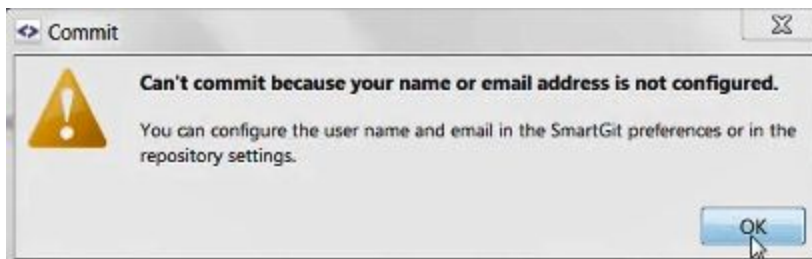


Все созданные файлы будут видны в программе SmartGit.

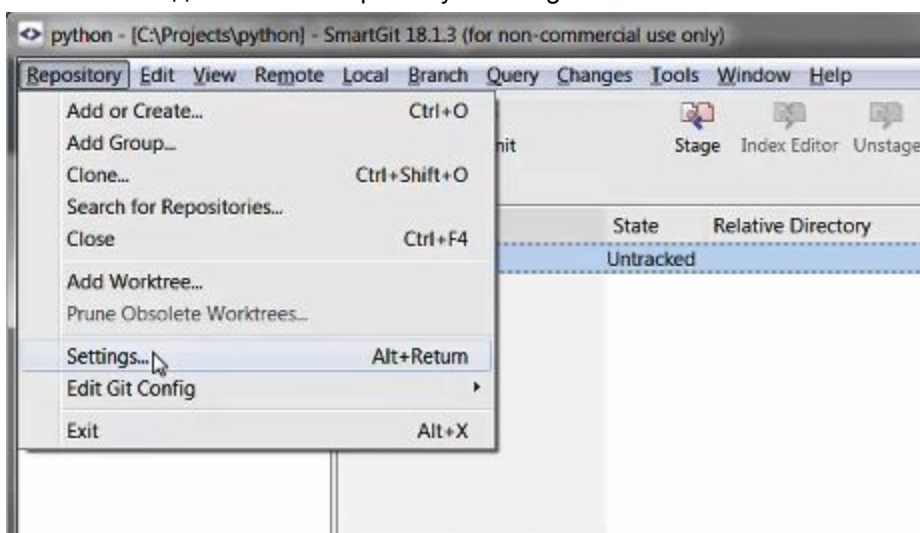
Для этих файлов нужно выполнить команду Commit, которая добавит их в локальный репозиторий.



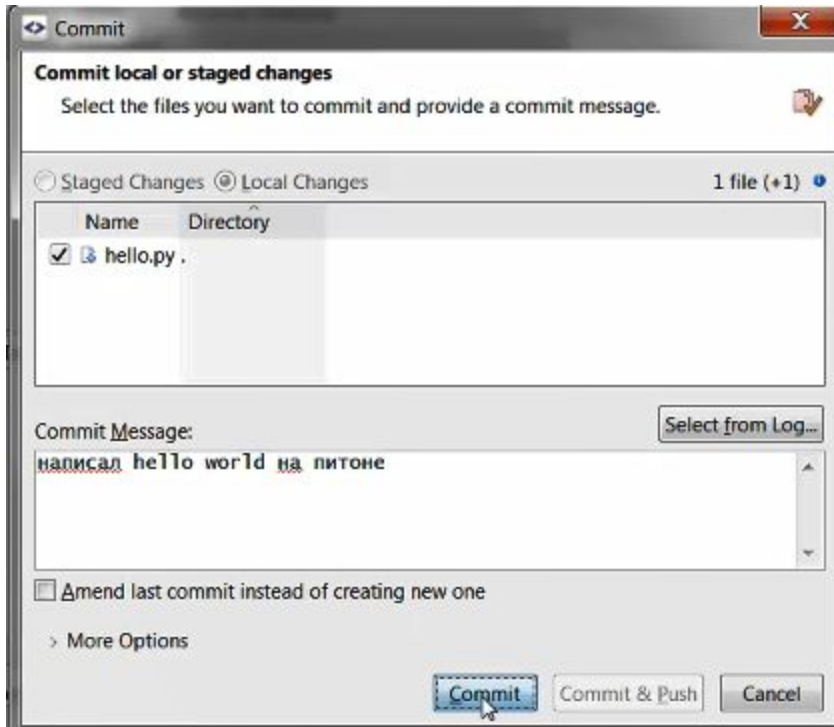
Возможно, при первом Commit-е вы увидите ошибку:



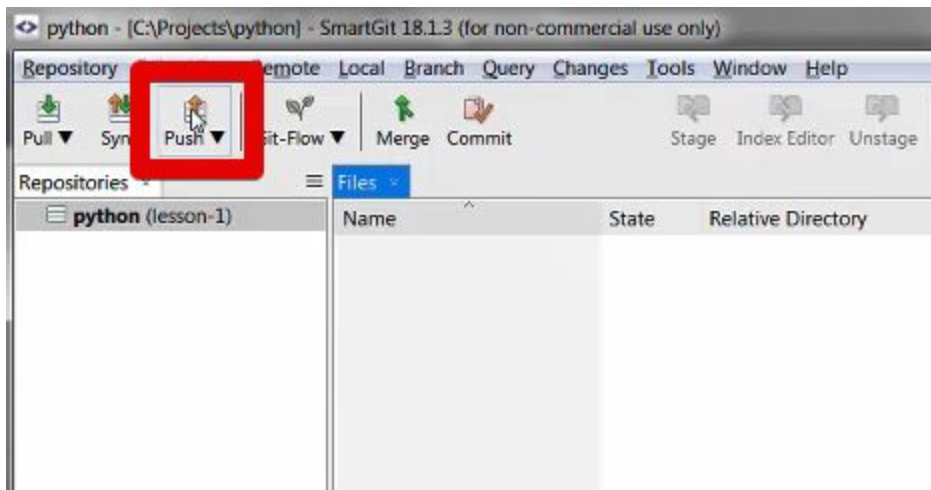
Без паники! Идем в меню Repository->Settings



Указываем свой Name и E-Mail. Нажимаем ОК. После этого повторяем команду Commit. При выполнении Commit-а потребуется указать комментарий.



Нажимаем Commit, затем Push:

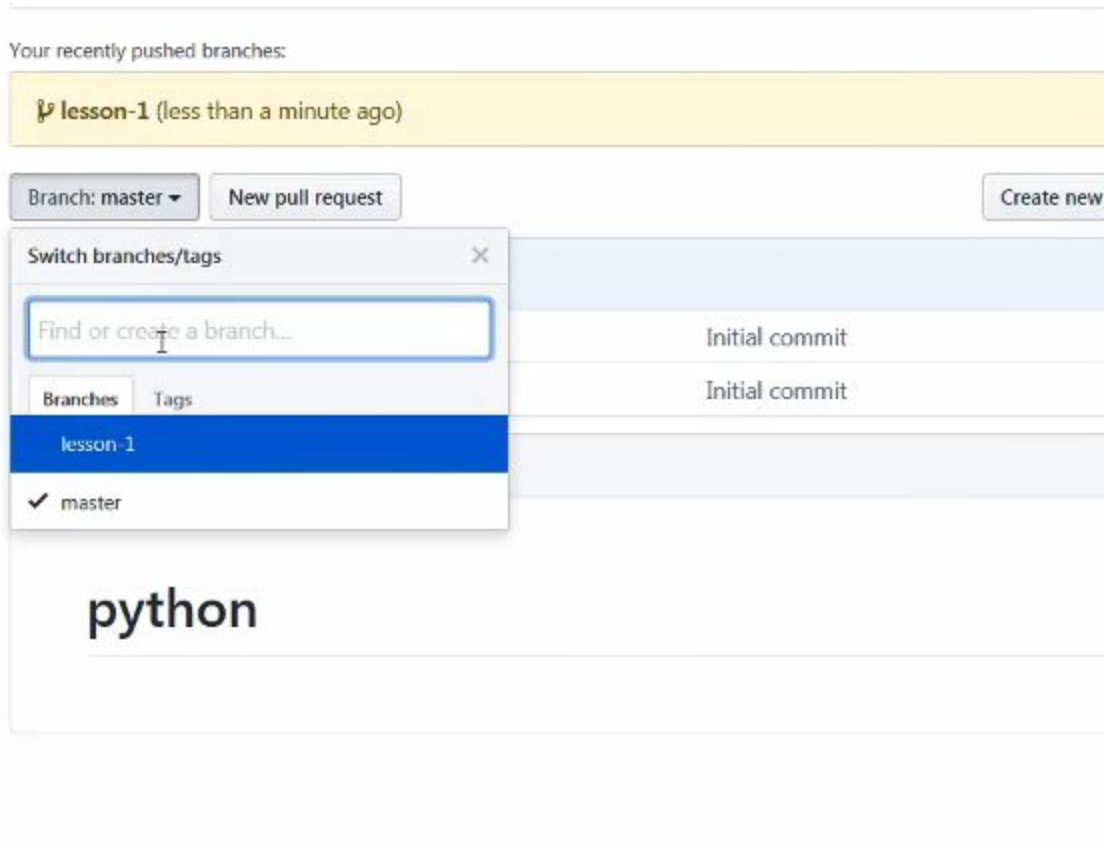


Выбираем опцию «Don't use master password», затем указываем свой логин и пароль на github.

После выполнения команды Push практическое задание отправлено на удаленный Git-репозиторий.

Создаем pull-request

Перейдите на сайте github на страницу вашего репозитория (в моем случае это <https://github.com/Kartavec/python>). Обновите страницу и перейдите в ветку lesson-1:



Вы должны увидеть файлы практического задания. Теперь нажимаем на кнопку New pull request:

Your recently pushed branches:

🔗 **lesson-1** (less than a minute ago)

Branch: lesson-1 ▾

New pull request

This branch is 1 commit ahead of master.

🔗 Kartavec написал hello world на питоне

📄 LICENSE Initial commit

📄 README.md Initial commit

📄 hello.py написал hello world на питоне

📄 README.md

При необходимости оставляем к практическому заданию комментарий – его увидит преподаватель – и нажимаем Create pull request:

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

🔗 base: master ▾

⬅️ compare: lesson-1 ▾

✓ **Able to merge.** These branches can be automatically merged.

🔗

написал hello world на питоне

Write

Preview

AA B i “ < > 🔗 ☰ ☷ ✓ @ 📌 ↶

Le

Здесь можно оставить комментарий

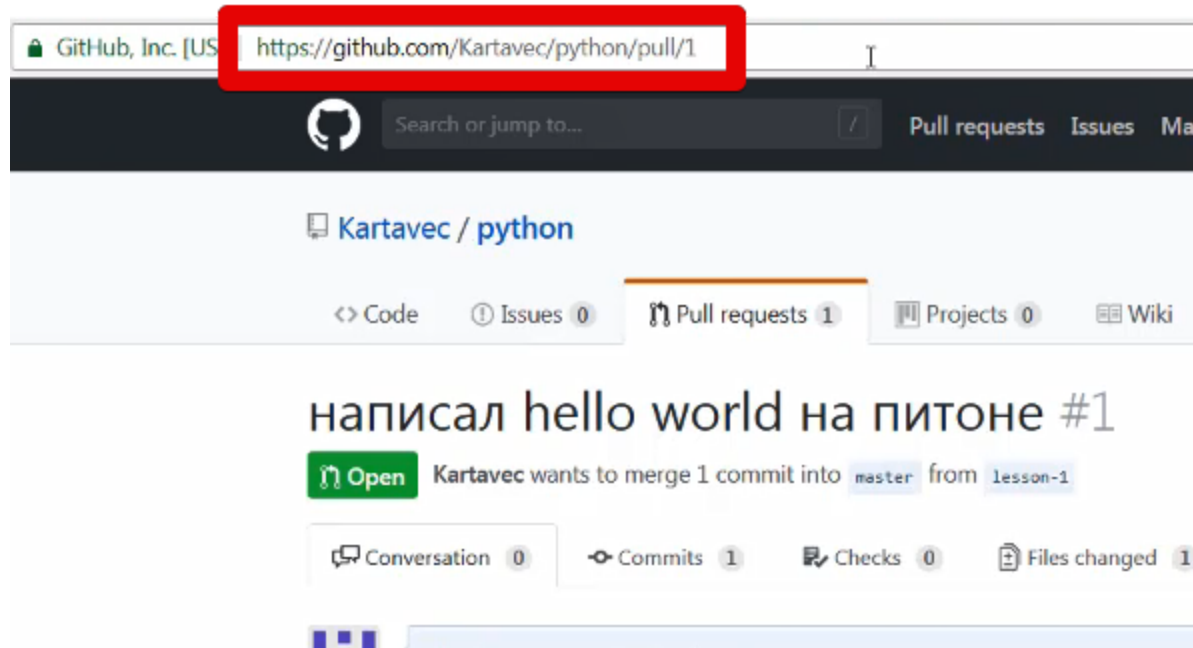
Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

📄 Styling with Markdown is supported

Create pull request

Сдача практического задания

Выполнив все описанные действия, вы перейдете по ссылке, которая будет вести на ваш pull-request. По этой ссылке преподаватель сможет проверить ваше практическое задание.



Сдайте эту ссылку в качестве практического задания на странице урока и нажмите «Сохранить».

Срок сдачи до: 20.08.2018 16:00 MSK

Если у вас возникли какие-то вопросы, вы всегда можете обратиться к своему наставнику: [Илья Круглов](#)

 <https://github.com/Kartavec/python/pull/1>

Сохранить

Чтобы научиться работать с Git, пройдите курс [Git. Быстрый старт](#)



Комментарий к домашней работе или ссылка на github-репозиторий, если вам не удалось создать pull-request

Домашнее задание должно быть сдано за 4 часа до начала следующего урока.

Практическое задание сдано! Преподаватель проверит его и прокомментирует прямо на сайте github на странице вашего pull-request-a.

Вместо эпилога

Загляните на вкладку “Files changed”. Именно в таком виде ваш код увидит преподаватель. Будет здорово, если там нет красно-зелёной зебры.

The screenshot shows a GitHub pull request for the repository `GeekBrainsTutorial / Python_lessons_basic`, which is a fork of `NADIP-Examples/Python_lessons_basic`. The repository has 4 watches, 10 stars, and 244 forks. The pull request is titled "Home Work1" and is from a user (represented by a grey box) to the `GeekBrainsTutorial:master` branch. It contains 4 commits. The "Files changed" tab is selected and highlighted with a red rectangle, showing 4 files changed. The diff view shows changes to the file `lesson01/home_work/hw01_easy.py`. The diff includes comments in Russian and Python code. The changes are highlighted with alternating green and red backgrounds, creating a "zebra" pattern. The code includes a task description and a solution using a `while` loop to process a number digit by digit.

GeekBrainsTutorial / Python_lessons_basic
forked from NADIP-Examples/Python_lessons_basic

Watch 4 Star 10 Fork 244

Code Pull requests 247 Projects 0 Wiki Insights

Home Work1

Open wants to merge 4 commits into GeekBrainsTutorial:master from :master

Conversation 0 Commits 4 Checks 0 Files changed 4

Changes from all commits Jump to... +117 -4

Diff settings Review changes

```
25 lesson01/home_work/hw01_easy.py
... @@ -1,13 +1,17 @@
1 1
2 - __author__ = 'Ваши Ф.И.О.'
2 + __author__ = 
3 3
4 4 # Задача-1: Дано произвольное целое число (число заранее неизвестно).
5 5 # Вывести поочередно цифры исходного числа (порядок вывода цифр неважен).
6 6 # Подсказки:
7 7 # * постарайтесь решить задачу с применением арифметики и цикла while;
8 8 # * при желании решите задачу с применением цикла for.
9 9
10 - # код пишем тут...
10 + value = input('Введите число: ')
11 + i = 0
12 + while i < len(value):
13 +     print(value[i])
14 +     i += 1
11 15
12 16
13 17 # Задача-2: Исходные значения двух переменных запросить у пользователя.
@@ -18,7 +22,24 @@
18 22 # Не нужно решать задачу так:
19 23 # print("a = ", b, "b = ", a) - это неправильное решение!
20 24
```