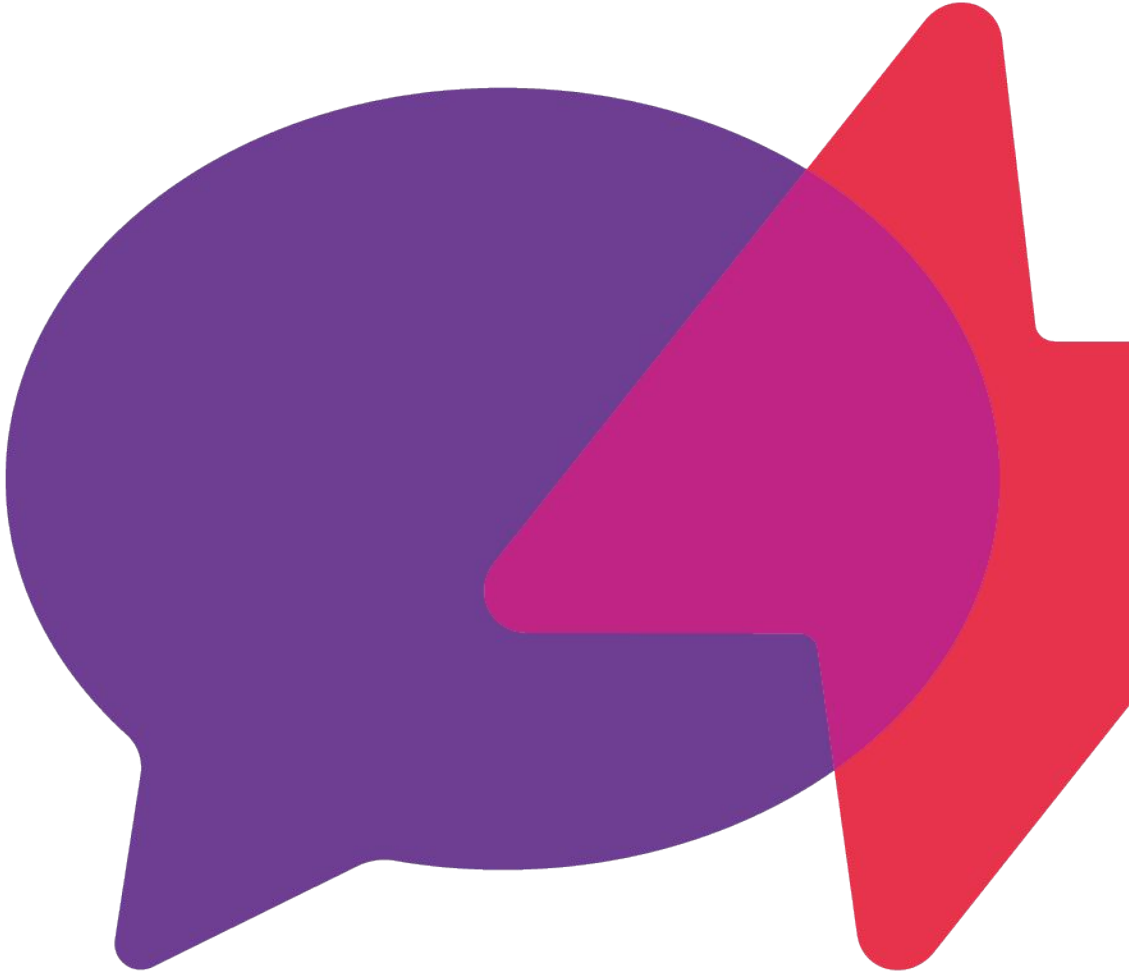
МЕЖДУНАРОДНАЯ
ШКОЛА
ПРОФЕССИЙ

Курсы Python для аналитиков



Система контроля версий Git

Занятие 9

Этапы занятия

- Что такое Git и для чего он создан
- Преимущества Git
- Сервис онлайн-хостинга репозиториев GitHub
- Интерфейс GitHub и базовый функционал

GIT -

это консольная утилита, для отслеживания и ведения истории изменения файлов, в вашем проекте.

Этап #1 - что такое Git и для чего он создан

Git — это инструмент совместного создания кода

Часто бывает так: разработчики отделяются от master-ветки и работают над частью проекта самостоятельно — например, чтобы протестировать дополнительные функции. Но не могут продолжить, пока кто-то из команды не допишет код.

Этап #1 - что такое Git и для чего он создан

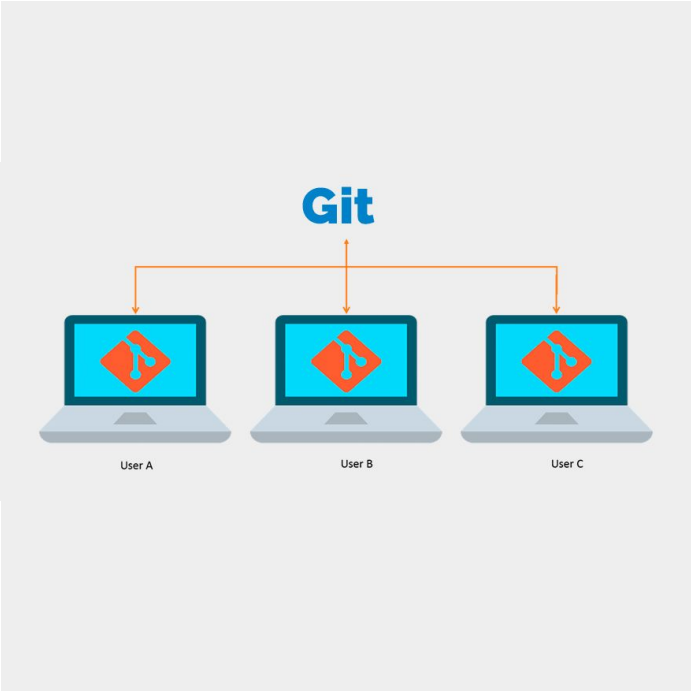
Система контроля версий позволяет
не ждать обновления master-ветки
и разрешает всем участникам команды
свободно перемещаться между ветками
других разработчиков для копирования
нужных фрагментов кода.

Этап #1 - что такое Git и для чего он создан

Git — это распределённая система версий

Системы контроля версий бывают локальными, централизованными или распределёнными.

Локальная система хранит файлы на одном устройстве, **централизованная** использует общий сервер, а **распределённая** — общее облачное хранилище и локальные устройства участников команды.



Этап #1 - что такое Git и для чего он создан

1. **В локальной системе** удобно работать с большими проектами, но сложно взаимодействовать с удалённой командой.

Этап #1 - что такое Git и для чего он создан

2 **В централизованной системе** налажена удалённая работа, но всё привязано к одному серверу. Любой сбой или взлом может повредить файлы проекта.

Этап #1 - что такое Git и для чего он создан

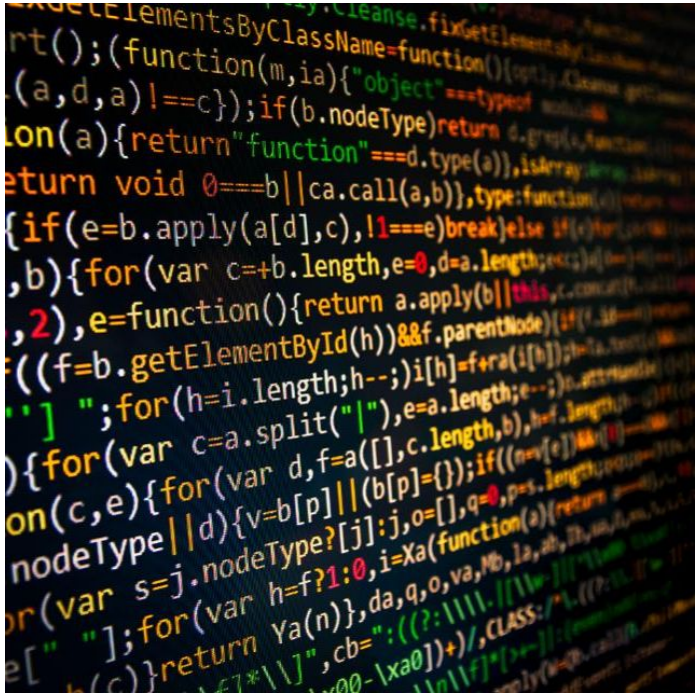
3 **В распределённой системе** налажена удалённая работа. Если с файлами основного репозитория что-то случится — проект легко восстановить из копии любого участника команды.

Этап #1 - что такое Git и для чего он создан

Git — это это система коммитов

В программировании за сохранение кода в контрольных точках отвечает **система контроля версий** — специальная технология, которую можно подключить к любому проекту.

Система контроля версий страхует от ошибок и возвращает код в то состояние, когда всё работало.



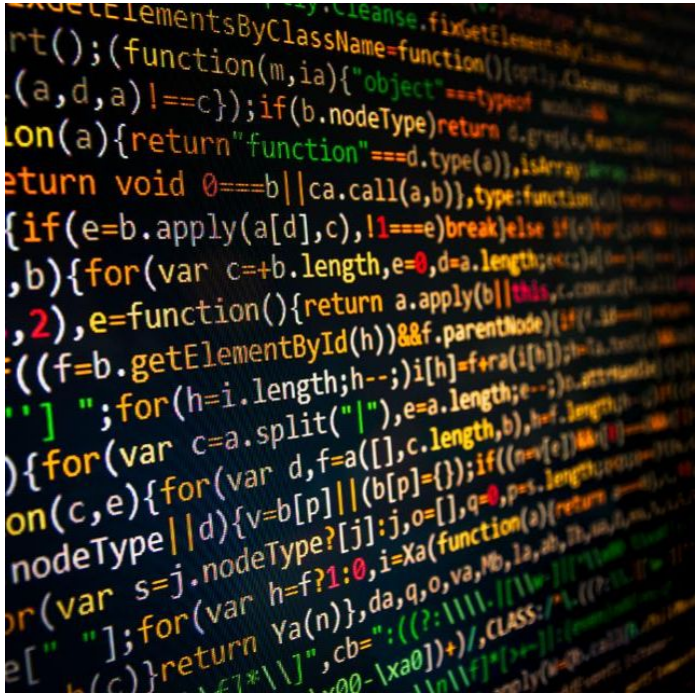
Этап #1 - что такое Git и для чего он создан

Git — это это система коммитов

Контрольные точки называются **коммитами.**

*Один **коммит** — это пакет изменений, хранящий информацию с добавленными, отредактированными или удалёнными файлами кода.*

В один коммит принято добавлять не более десяти изменений — так получается длинная история версий, которая позволяет в случае ошибки откатиться с минимальной потерей работоспособного кода.

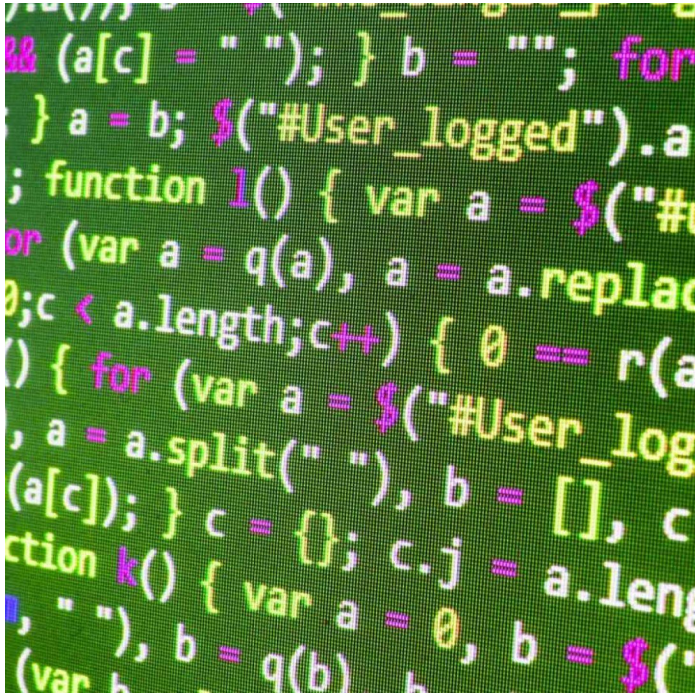


Этап #1 - что такое Git и для чего он создан

Git — это комплекс связанных веток

Коммиты располагаются на **master-ветке** — основной версии проекта, которая после завершения работы превратится в продукт.

Система контроля версий позволяет создавать ответвления от master-ветки и экспериментировать с проектом, не мешая другим участника команды.



Этап #2 - преимущества Git

1. Гибкая система ветвления проектов и слияния веток между собой.
2. Наличие локального репозитория, содержащего полную информацию обо всех изменениях, позволяет вести полноценный локальный контроль версий и заливать в главный репозиторий только полностью прошедшие проверку изменения.

Этап #2 - преимущества Git

3. Высокая производительность и скорость работы.
4. Возможность делать контрольные точки. Это позволяет уменьшить скорость восстановления данных, так как за основу берется ближайшая контрольная точка, и восстановление идет от нее.

   Если бы контрольные точки отсутствовали, то восстановление больших проектов могло бы занимать часы.

Этап #2 - преимущества Git

5. Широкая распространенность и легкая доступность.
6. Гибкость системы позволяет удобно ее настраивать и даже создавать специализированные контроля системы или пользовательские интерфейсы на базе git.

Этап #3 - Сервис онлайн-хостинга репозиториев GitHub

Если ваш сайт — это статические HTML-страницы, то необязательно приобретать хостинг, можно воспользоваться сервисом GitHub Pages.

Для этого у вас должен быть аккаунт на Гитхабе.

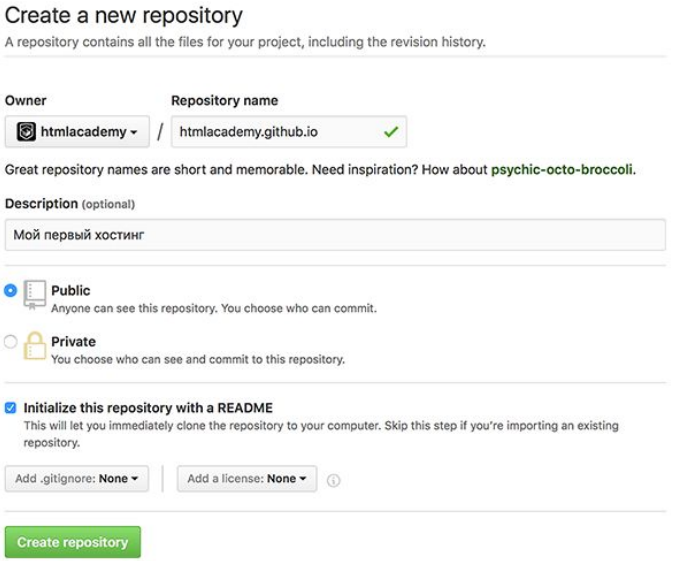
Здесь вы можете хранить свои файлы для команды или же содержать свой статический сайт.

Шаг 1. Создание нового репозитория

Для создания репозитория заходим на сайт Гитхаба и в блоке **«Your repositories»** нажимаем кнопку **«New repository».**

Теперь нам нужно заполнить параметры нового репозитория. Важно, чтобы название репозитория было в виде **«username.github.io»,** где username — имя вашего аккаунта на Гитхабе.

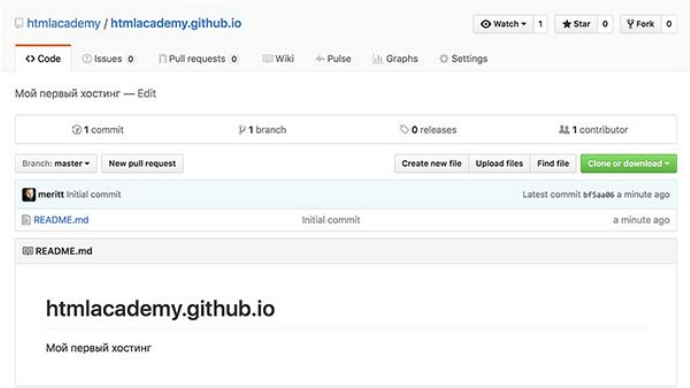
В нашем примере это будет «htmlacademy.github.io».



Шаг 2. Загрузка файлов

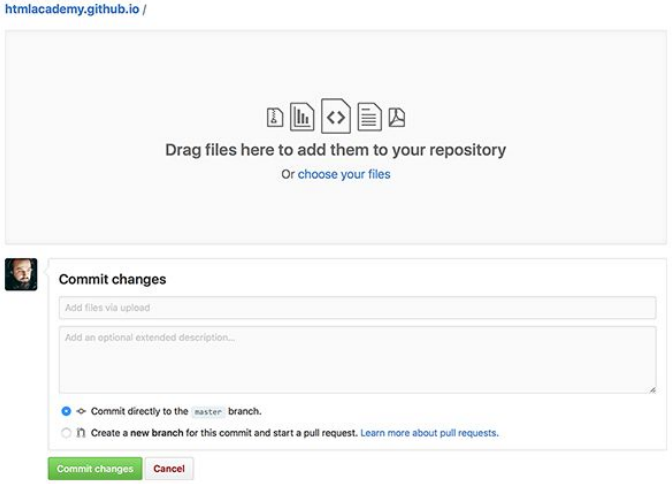
Обязательно установим галочку **«Initialize this repository with a README».** А затем нажмём кнопку **«Create repository».**

Репозиторий создан, теперь нужно загрузить файлы. Для этого необязательно клонировать репозиторий к себе на компьютер или постигать другие нюансы работы с Гитом — можно воспользоваться интерфейсом Гитхаба.
Давайте нажмём кнопку **«Upload files».**



Шаг 2. Загрузка файлов

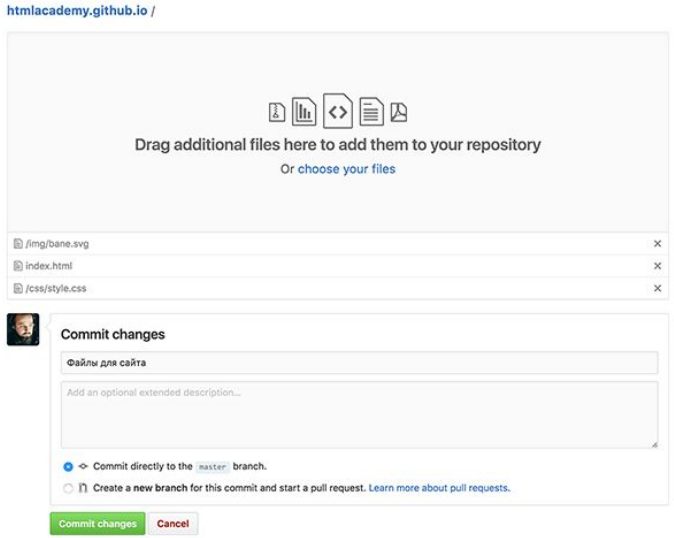
Затем **перетащим файлы** в появившееся поле для загрузки.



Шаг 2. Загрузка файлов

Добавим комментарий к нашему коммиту и нажмём кнопку **«Commit changes».**

Готово! Файлы загружены в репозиторий.



Шаг 3. Проверка работы сайта

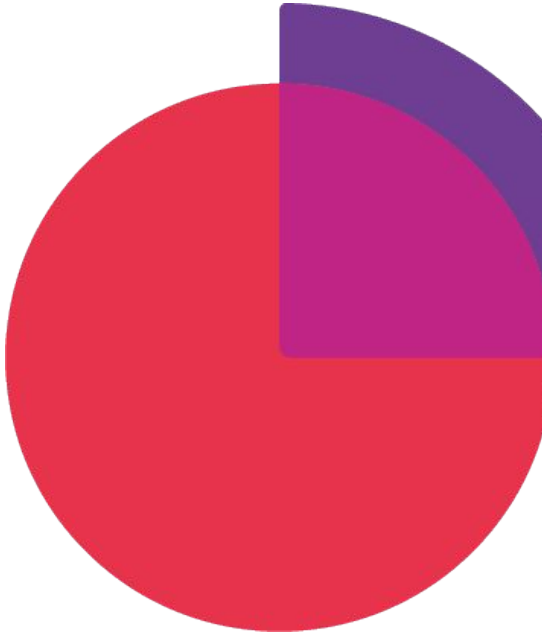
Чтобы проверить работу сайта, достаточно перейти по адресу **username.github.io,** в нашем случае это **htmlacademy.github.io.**

Этап #4 - интерфейс GitHub и базовый функционал

Установка

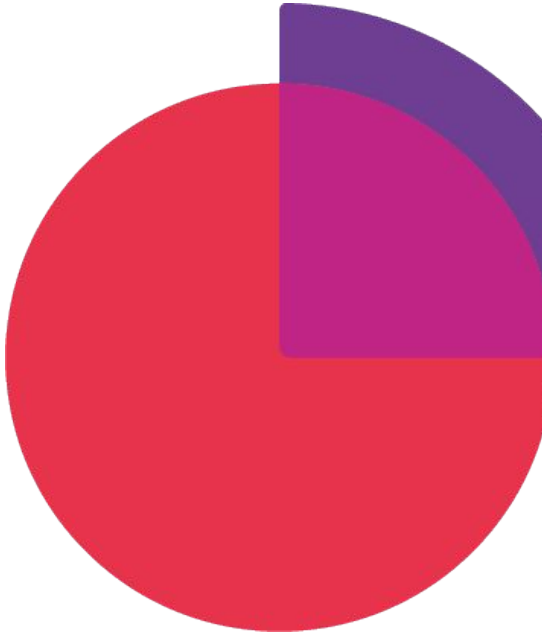
Основой интерфейс для работы с Git-ом является консоль/терминал. Это не совсем удобно, тем более для новичков, однако это дело привычки.

Этап #4 - интерфейс GitHub и базовый функционал



Windows: Проходим по этой ссылке: https://git-scm.com/download/win, выбираем под вашу ОС (32 или 64 битную), скачиваем и устанавливаем.

Этап #4 - интерфейс GitHub и базовый функционал



Для Mac OS: Открываем терминал и пишем:

#Если установлен Homebrew

brew install git

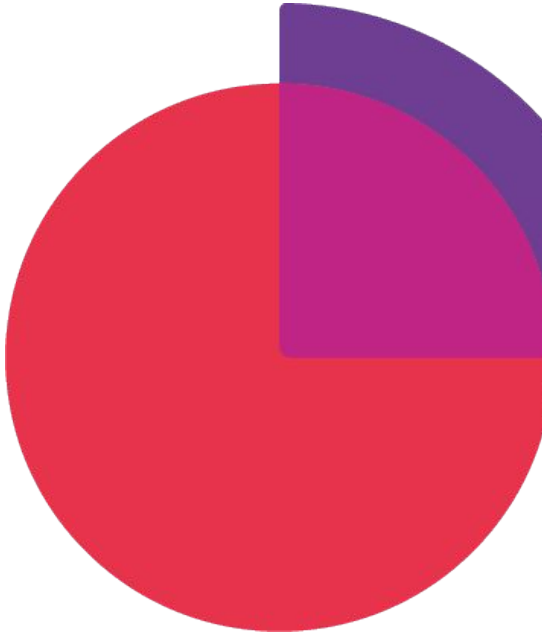
#Если нет, то вводим эту команду.

git --version

#После этого появится окно, где предложит установить Command Line Tools (CLT).

#Соглашаемся и ждем установки. Вместе с CLT установится и git

Этап #4 - интерфейс GitHub и базовый функционал



Linux: Открываем терминал и вводим следующую команду

 # Debian или Ubuntu

sudo apt install git

 # CentOS

sudo yum install git

Настройка

Вы установили себе Git и можете им пользоваться. Настроим его, чтобы при создании commit, указывался автор, кто его создал. **Вводим следующее:**

#Установим имя для вашего пользователя

#Вместо <ваше_имя> можно ввести, например, *Grisha_Popov*

#Кавычки оставляем

git config --global user.name "<ваше_имя>"

#Теперь установим email по тому же принципу

git config --global user.email "<адрес_почты@email.com>"

Создание репозитория

Теперь вы готовы к работе с Git локально на компьютере.

Создадим наш первый репозиторий. Для этого пройдите в папку вашего проекта.

#Для Linux и MacOS путь может выглядеть так:
/Users/UserName/Desktop/MyProject

#Для Windows, например, C://MyProject
cd <путь_к_вашему_проекту>

#Инициализация/создание репозитория

git init

Создание репозитория

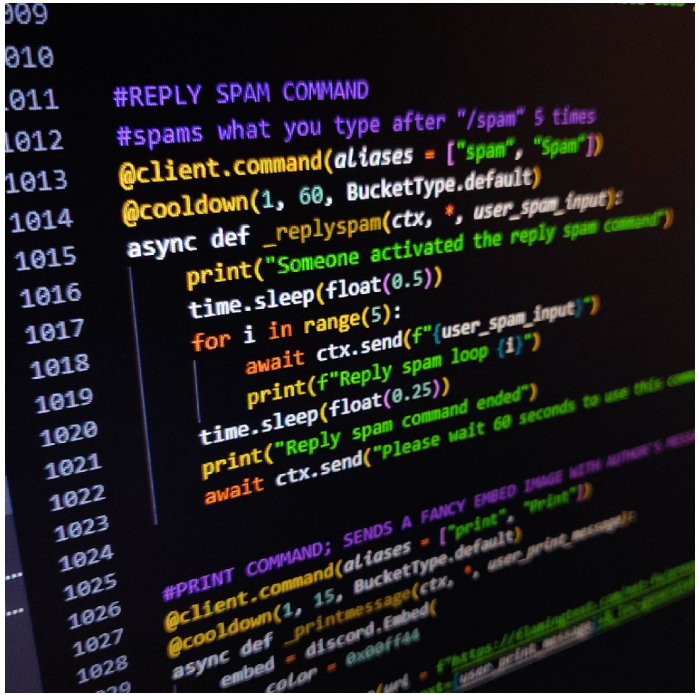
Теперь Git отслеживает изменения файлов вашего проекта. Но, так как вы только создали репозиторий в нем нет вашего кода.
Для этого необходимо **создать commit.**

#Добавим все файлы проекта в наш будущий commit

git add

#Или так

git add --all



Создание репозитория

#Если хотим добавить конкретный файл, то можно так:

git add <имя_файла>

#Теперь создаем commit. Обязательно указываем комментарий.

#И не забываем про кавычки

git commit -m "<комментарий>"

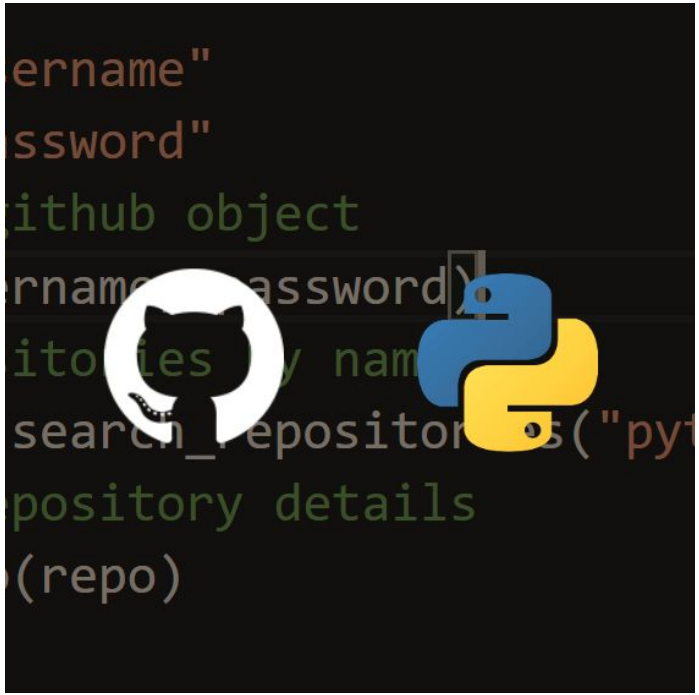
Отлично. **Вы создали свой первый репозиторий и заполнили его первым commit.**

Процесс работы с Git

Не стоит после каждого изменения файла делать commit.
Чаще всего **их создают, когда:**

- Создан новый функционал
- Добавлен новый блок на верстке
- Исправлены ошибки по коду
- Вы завершили рабочий день и хотите сохранить код

Это поможет держать вашу ветки в чистоте и порядке.
Тем самым, вы будете видеть историю изменений
по каждому нововведению в вашем проекте,
а не по каждому файлу.



Полезные инструменты в работе

- **Официальный сайт с командами:**

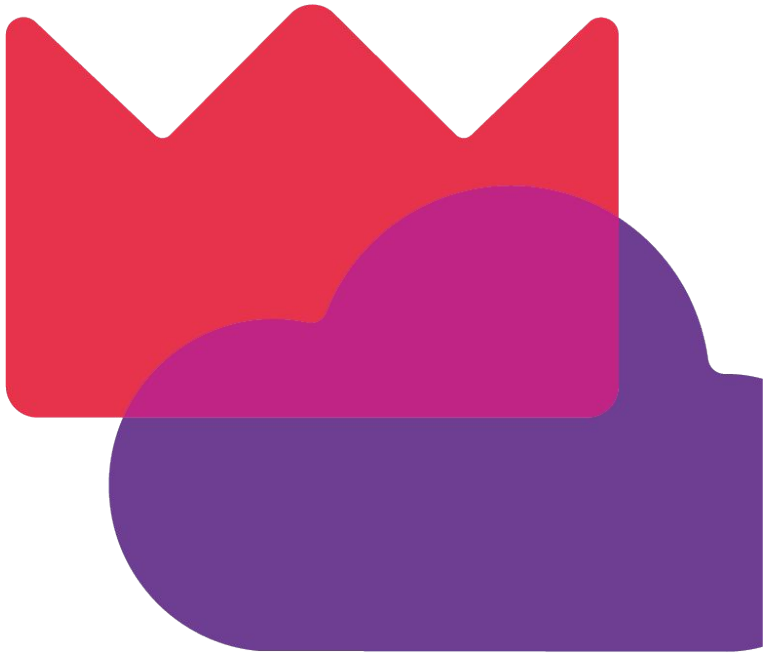
  https://gist.github.com/rdnvndr/cb21a06c5a71fd71213aed1619380b8e

- **Для работы с кодом и github лучше использовать Visual Studio Code:**

  https://code.visualstudio.com/download

Домашнее задание:

1. Создайте свою репозиторию на онлайн-хостинге github, загрузите туда пару своих файлов, а затем скиньте ссылку на github.



Теперь вы знаете:

- Что такое Git и для чего он создан
- Преимущества Git
- Сервис онлайн-хостинга репозиториев GitHub
- Интерфейс GitHub и базовый функционал