Git и GitHub | Я.Шпора

Настройка Git

Если Git установлен

Проверить наличие Git на компьютере:

```
git --version
```

Если Git установлен, выведется подобное сообщение:

```
git version 2.30.0
```

Если Git не установлен

Git можно установить при помощи инсталлятора с официального сайта или через пакетный менеджер, например Homebrew для macOS или APT для Linux.

Базовые настройки Git

В программе укажите ваше имя. Откройте Git Bash на Windows или терминал на Linux/macOS и выполните команду:

```
git config --global user.name "Имя Фамилия"
```

Это имя будет применяться для всех репозиториев на вашем компьютере. Можно проверить установленное ранее значение с помощью команды:

```
$ git config --global user.name
```

Далее укажите ваш действующий адрес электронной почты:

```
git config --global user.email "stas.basov@whatever.com"
```

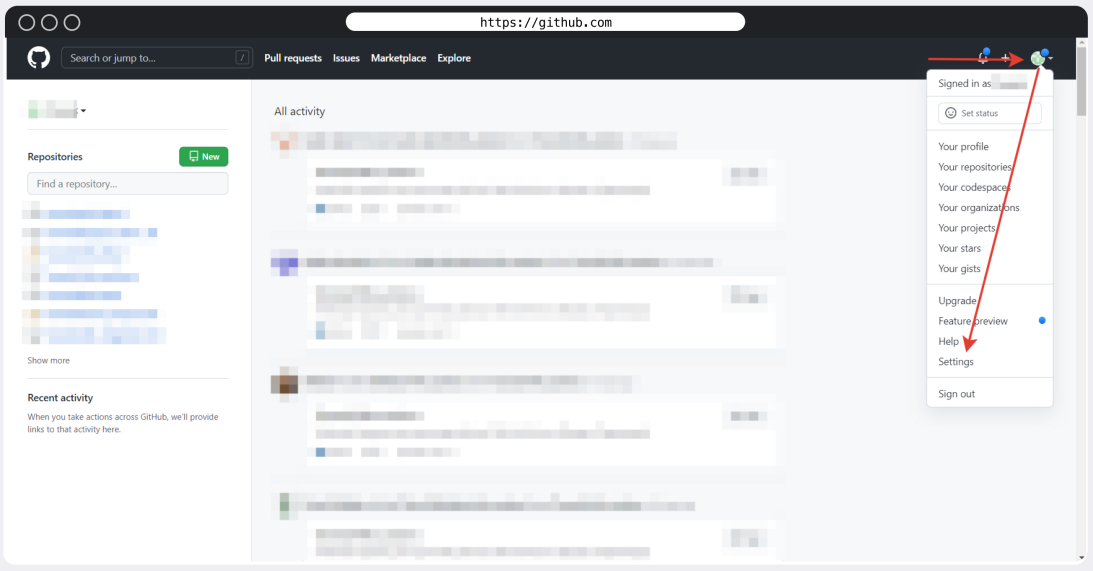
> Эту конфигурацию нужно выполнить только один раз при первой настройке Git.

Настройка GitHub

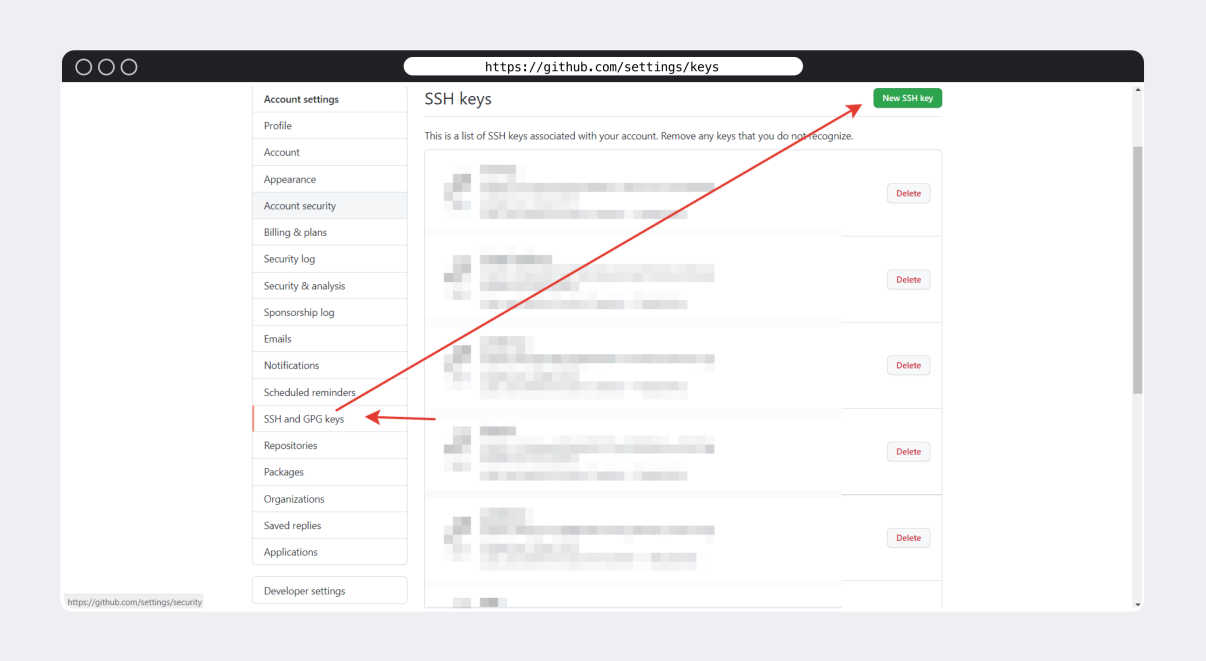
Регистрация в веб-сервисе GitHub осуществляется на сайте. Для регистрации нужно указать свой электронный адрес, придумать пароль и задать имя пользователя. Далее следует подтвердить регистрацию, используя код из письма, которое придёт на указанную почту.

Настройка связи между компьютером и GitHub: SSH-ключи

1. Запустите Git Bash на Windows или терминал на Linux/macOS. Выполните команду `ssh-keygen` .
2. Консоль попросит ввести путь к файлу, в который будут сохранены сгенерированные ключи, и одновременно предложит сохранить их в файл по умолчанию.
3. Сохраните ключи в папку по умолчанию: для этого нажмите **Enter** на Windows или **Return** на macOS.
4. При создании ключей система попросит придумать пароль для доступа к ключам. Когда вы будете вводить пароль, в терминале ничего не отобразится, даже звёздочки.
5. Выведите открытый ключ в терминал командой `cat .ssh/id_rsa.pub` .
6. Скопируйте ключ от символов `ssh-rsa` включительно и до конца.
7. Зайдите в свой аккаунт на GitHub, перейдите в раздел настроек:



8. Выберите пункт **SSH and GPG keys**; для создания нового ключа нажмите на кнопку **New SSH key** в правом верхнему углу:



9. Откроется страница с двумя полями ввода. Введите любое имя и вставьте в поле **Key** ключ из буфера обмена.
10. Нажмите кнопку **Add SSH key** — ключ добавится к вашему аккаунту.

> Если вы захотите получить SSH-доступ к своему аккаунту на GitHub с нескольких компьютеров, для каждого из них должен быть создан и добавлен свой SSH-ключ.

Создание и клонирование репозиториев

Как создать локальный Git-репозиторий

1. Откройте терминал и перейдите в директорию с вашим проектом:

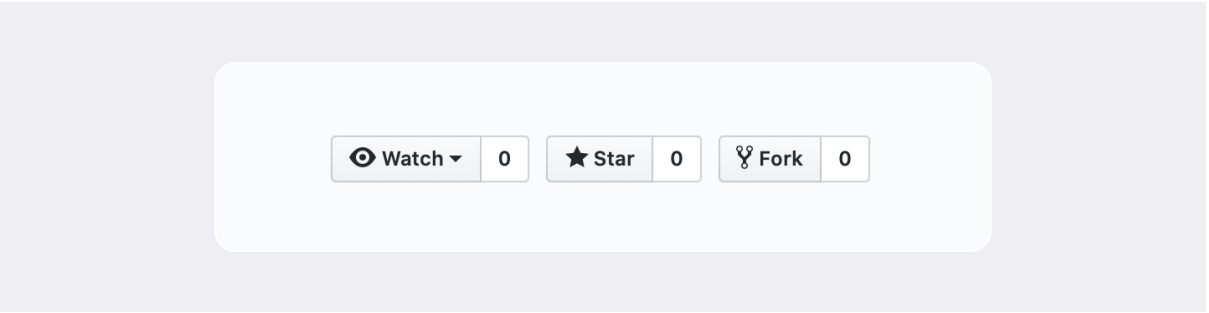
```
cd /Dev/first_project
```

2. Инициализируйте новый Git-репозиторий командой:

```
git init
```

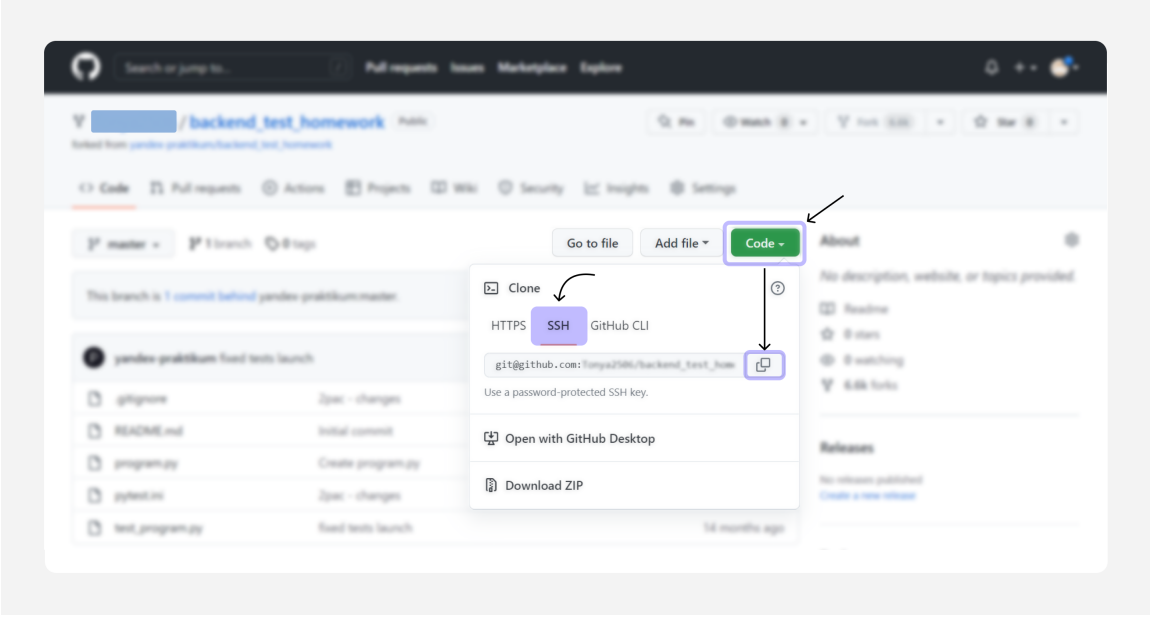
Как скопировать репозиторий

Операция копирования чужого репозитория в свой аккаунт называется «форк», этот процесс происходит на сайте GitHub. Чтобы форкнуть репозиторий, нужно зайти в него на сайте GitHub и нажать кнопку **Fork**:



Как клонировать репозиторий

1. На GitHub перейдите в репозиторий, который хотите клонировать, и скопируйте ссылку на него со вкладки SSH:



2. На компьютере перейдите в директорию, куда хотите клонировать репозиторий, с помощью команды `cd` .
3. Клонируйте репозиторий:

```
git clone <ссылка_на_репозиторий>
```

Основные команды для работы с Git

Узнать состояние репозитория

```
git status
```

Добавить изменения в индекс

В конкретном файле:

```
git add <название_файла>
```

Во всех отслеживаемых файлах сразу:

```
git add --all
```

Опцию `--all` можно заменить точкой `.`.

Создать коммит и комментарий к нему

```
git commit -m "Add first commit"
```

Изменить последний коммит и обновить комментарий

```
git commit --amend -m "Текст вашего комментария"
```

Отправить коммит на GitHub

```
git push
```

Удалить ненужные файлы или директории из репозитория

```
git rm file1.txt
```

Чтобы зафиксировать изменения, необходимо сделать коммит после удаления.

Просмотреть историю изменений

```
git log
```

Просмотреть изменения, внесённые в определённом коммите

```
git show id_коммита
```

> id_коммита — первые семь символов идентификатора нужного коммита.

Вернуться к определённому коммиту: откатить изменения

```
git reset id_коммита
```

Откатиться на один коммит назад в определённом файле:

```
git reset HEAD program.py
```

HEAD указывает на то, что работа будет вестись с последним сделанным коммитом.

Если не указать имя файла, то в репозитории сбросятся все изменения до состояния последнего коммита.

Игнорирование файлов

Чтобы файлы не попали в репозиторий, нужно указать названия директорий или имена файлов в текстовом файле *.gitignore.* Этот файл создаётся в корневой директории проекта.

```
# Игнорировать файл README.md.
README.md

# Игнорировать файл side.txt в директории build.
build/side.txt

# Игнорировать все файлы с расширением .doc
*.doc
```

Шпаргалка — это карта,

помогающая мудрому
мореплавателю выбрать путь.

Ягдетто Там, испанский мореплаватель