

---

# *Git*

---

Контроль версий

<https://git-scm.com/doc>  
Getting-Started-Git-Basics  
<https://git-scm.com/videos>

## НЕОБХОДИМОСТЬ КОНТРОЛЯ ВЕРСИЙ

- Вы всегда работаете в команде:
  - Ваши коллеги вносят правки
  - Вам-из-будущего будет очень сложно вспомнить, зачем вы-из-прошлого внесли конкретное изменение
- Сохранение – часто и надежно:
  - Меньше беспокойства
  - Последняя версия всегда актуальна...
  - ...но вся история доступна!
- Дополнительный эффект: всегда легко поделиться кодом

## УСТАНОВКА GIT

В терминале (рекомендованный вариант)

- MacOS and Linux

Доступен по умолчанию в терминале

- Windows

- <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
- (Как вариант можно использовать [GitBash](#))

- Без терминала

В большинстве IDE встроена функция использования Git (Pycharm, VSCode, Atom, ...) - Обратитесь к документации

- [GitKraken](#)

## БЫСТРОЕ КОНФИГУРИРОВАНИЕ

**\$ git --version**

**\$ git config --list**

**\$ git config --global user.name "yourName"**

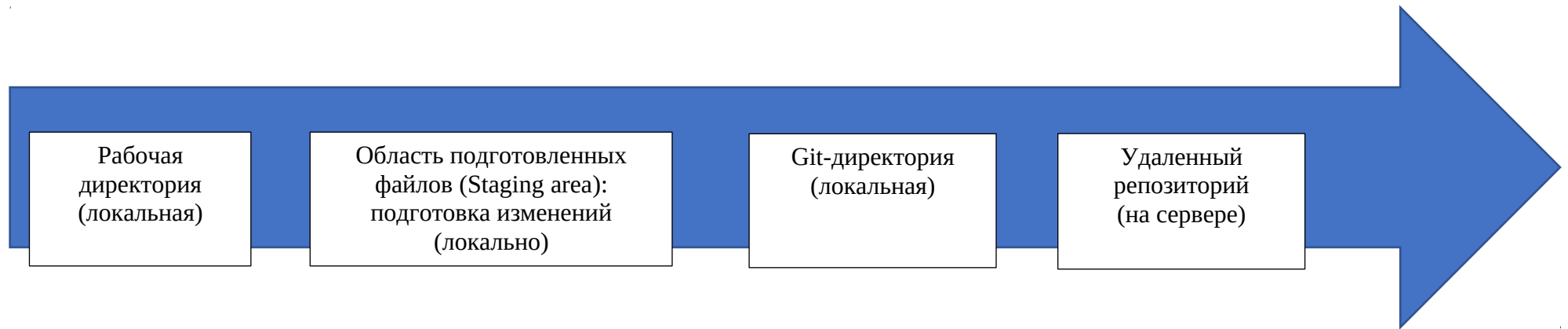
**\$ git config --global user.email "ns@mail.com"**

**\$ git config --global core.editor vim**

**> <https://help.github.com/articles/associating-text-editors-with-git/>**

**\$ git config --list**

## СТРУКТУРА GIT



Ваши локальные файлы, готовые к редактированию

Сюда вы перемещаете файлы после редактирования

Сюда **коммитятся** (**commit**) изменения.

Это создает локально новую версию файлов.


Изменения окончательно переносятся (**push**) на сервер, где они будут сохранены.

Известные провайдеры:




- [Github](#)
- [Gitlab](#)
- [BitBucket](#)
- [Cspark's Gitlab](#)

## Создание нового репозитория

На **Github**:



[Pulls](#) [Issues](#) [Marketplace](#) [Explore](#)



---



### Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

---

Owner \*

Repository name \*

/ free\_it\_py\_data\_analysis 



Great repository names are short and memorable. Need inspiration? How about [symmetrical-octo-pancake?](#)

Description (optional)

---

## СОЗДАНИЕ НОВОГО РЕПОЗИТОРИЯ

На **Github**:

- 
- ☐  **Public**  
Anyone on the internet can see this repository. You choose who can commit.
- ☒  **Private**  
You choose who can see and commit to this repository.
- 

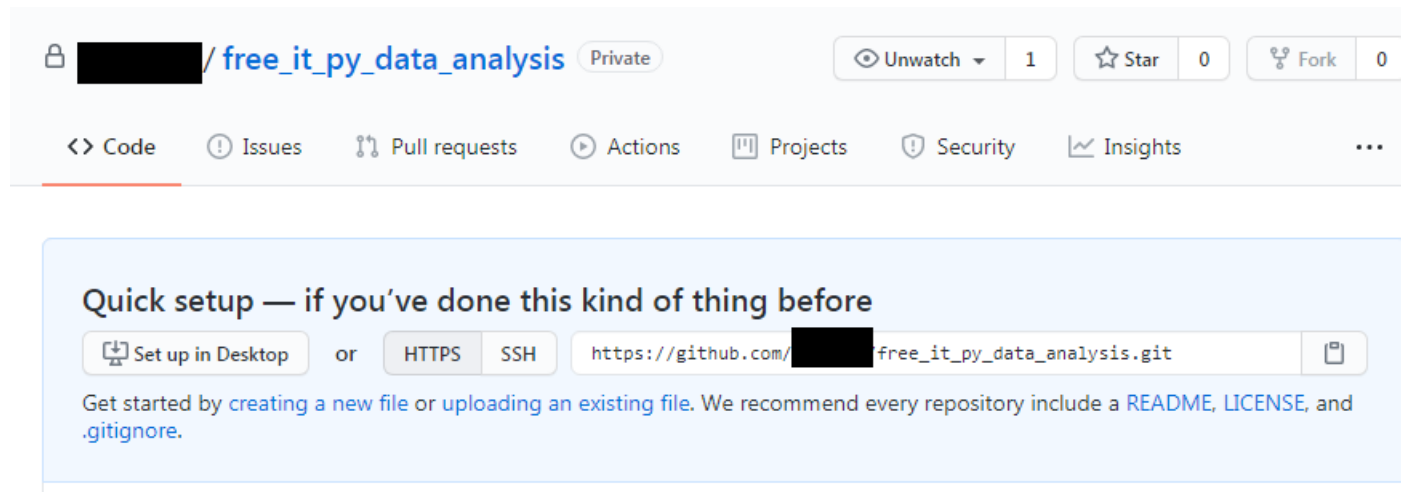
**Initialize this repository with:**

Skip this step if you're importing an existing repository.

- ☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)
- ☐ **Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)
- ☐ **Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)
- 

Create repository

## КЛОНИРОВАНИЕ РЕПОЗИТОРИЯ



Чтобы клонировать репозиторий, скопируйте предложенный url-адрес

Можно копировать через:

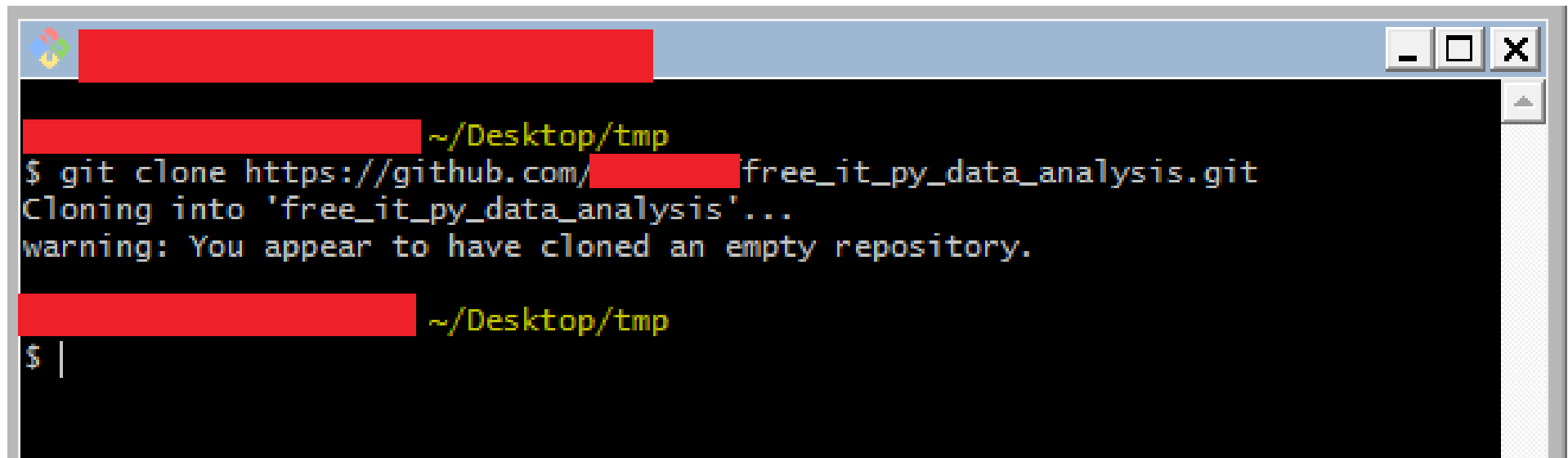
- **HTTPS:** Не требуется дополнительных настроек, но потребуется ввести ваш пароль
- **SSH:** Нужно будет один раз добавить ключ, затем аутентификация будет проходить автоматически



## КЛОНИРОВАНИЕ РЕПОЗИТОРИЯ

**\$ cd where/you/want/to/clone/the/repo**

**\$ git clone the/url/you/copied**

A screenshot of a terminal window with a blue title bar and standard window controls. The terminal has a black background with yellow and white text. It shows a user at the prompt cloning a repository. The output includes a warning about an empty repository. The user's name and the repository URL are redacted with red boxes.

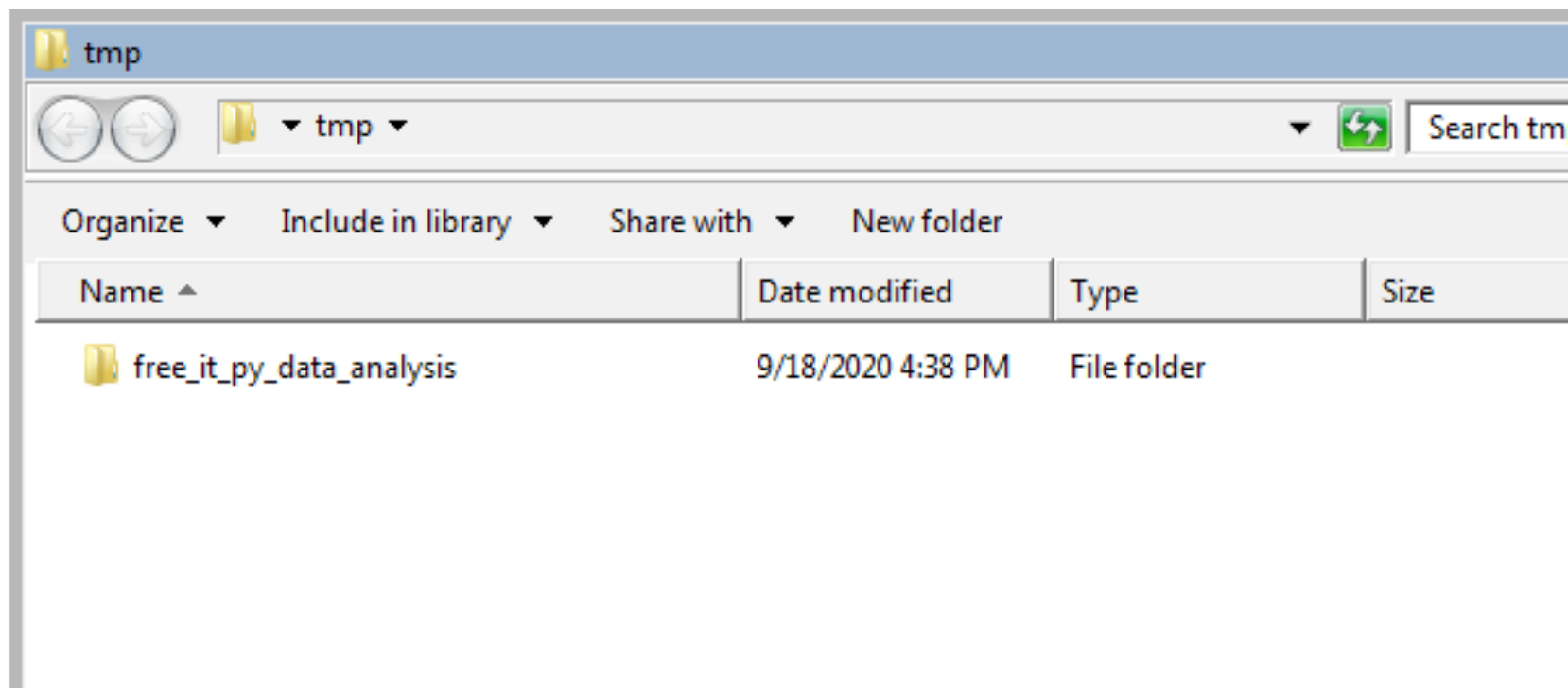
```
~/Desktop/tmp
$ git clone https://github.com/[redacted]free_it_py_data_analysis.git
Cloning into 'free_it_py_data_analysis'...
warning: You appear to have cloned an empty repository.

~/Desktop/tmp
$ |
```

## КЛОНИРОВАНИЕ РЕПОЗИТОРИЯ

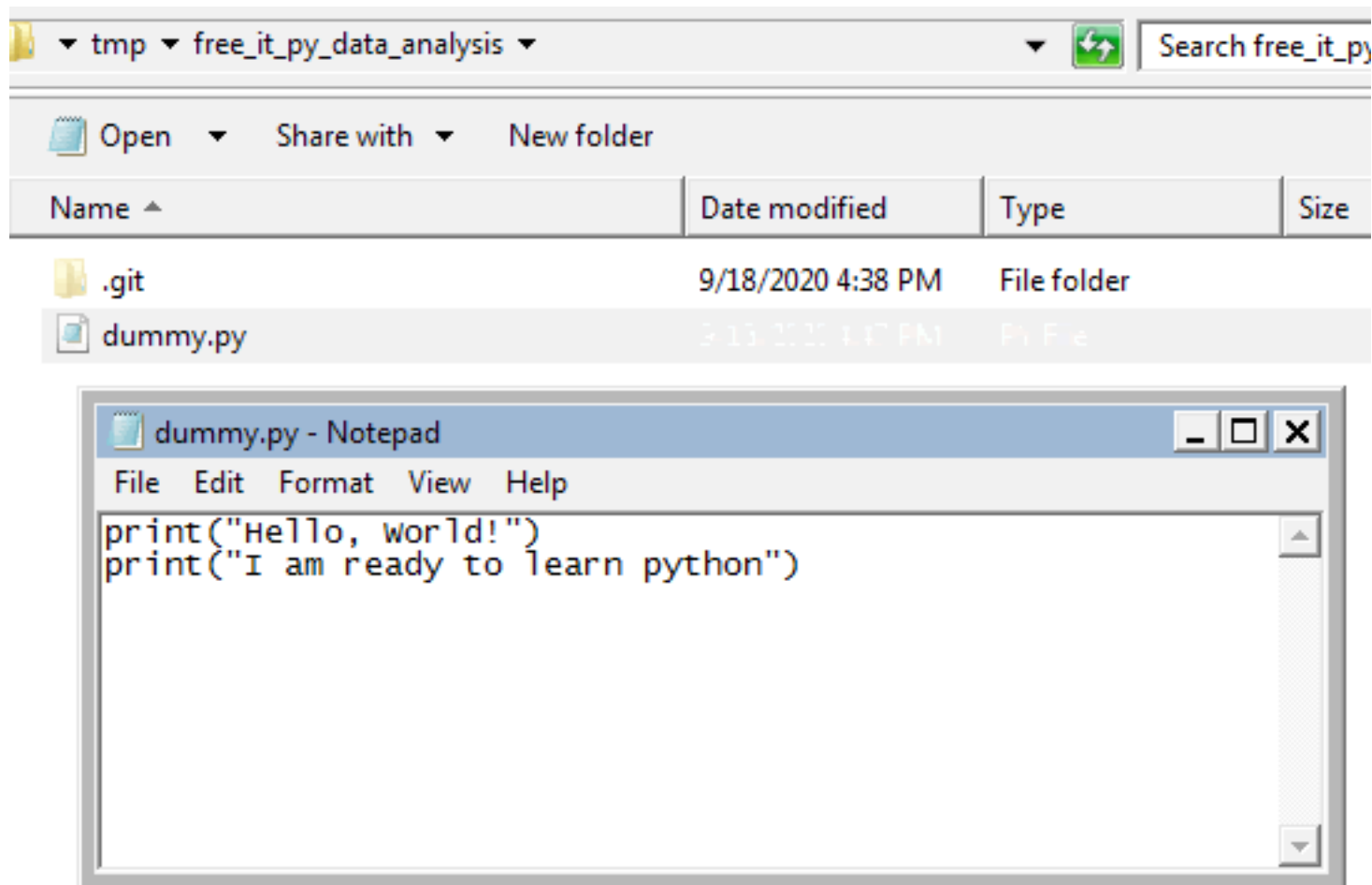
**\$ cd free\_it\_py\_data\_analysis**

**Когда репозиторий скопирован, можно начинать работу в нем, как в обычной папке**

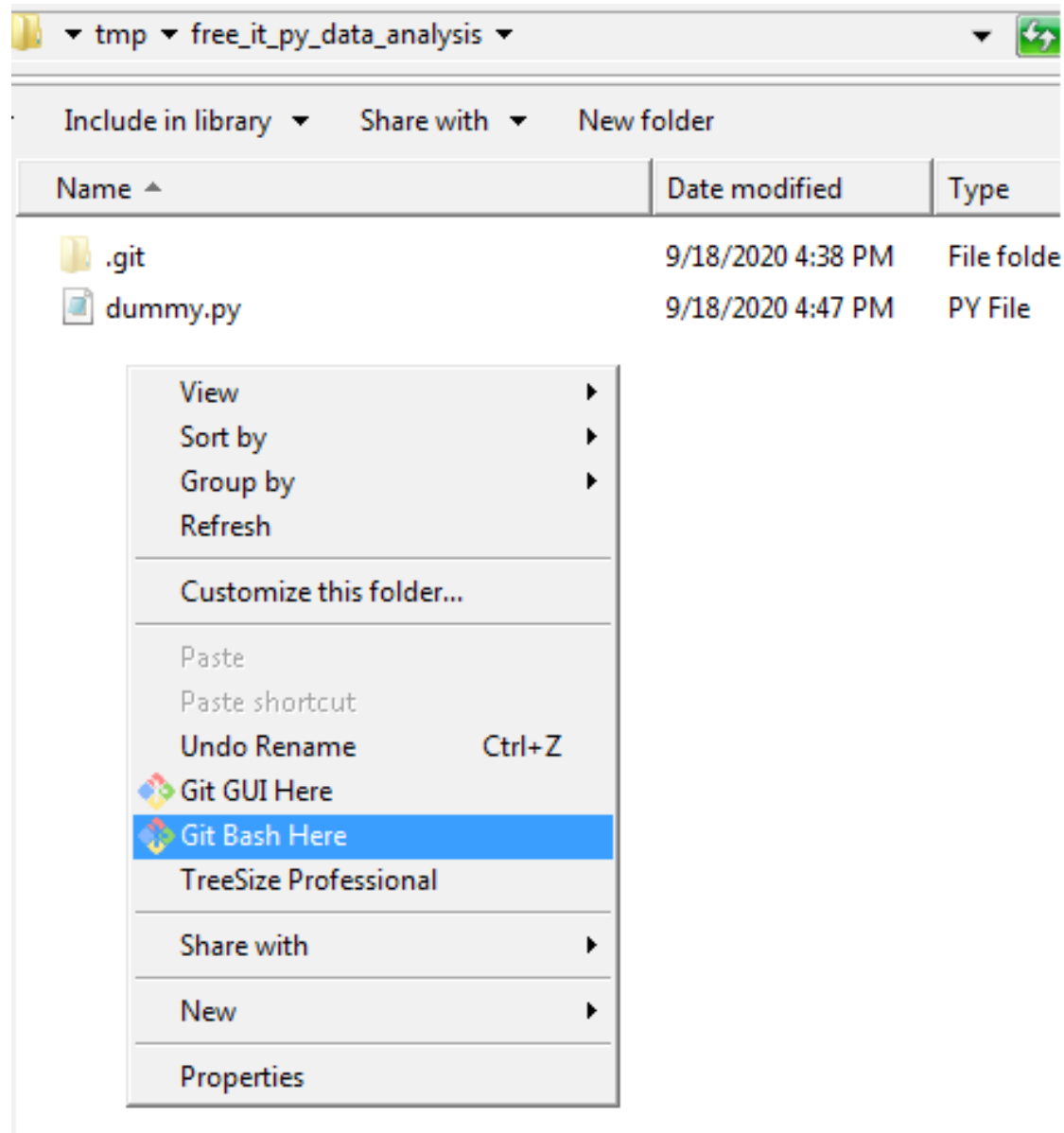


## СОХРАНЕНИЕ ИЗМЕНЕНИЙ

- Откройте **папку** и создайте новый файл **dummy.py**
- Напишите код и сохраните файл

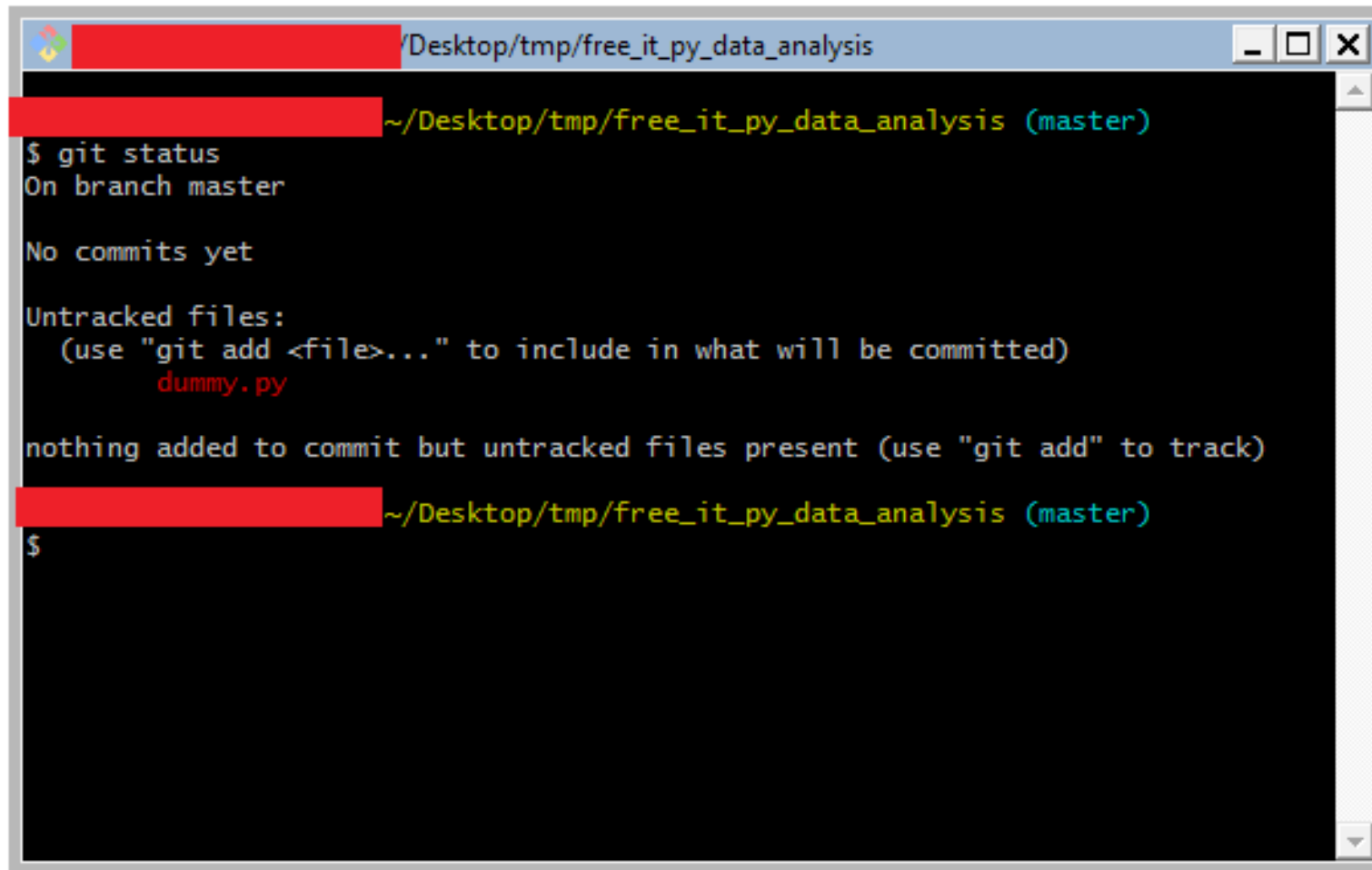


## OPEN GIT BASH



## СОХРАНЕНИЕ ИЗМЕНЕНИЙ В ОБЛАСТЬ ПОДГОТОВЛЕННЫХ ФАЙЛОВ (STAGING AREA)

**\$ git status** #чтобы увидеть файлы в рабочей папке и в staging area

A screenshot of a Windows command prompt window. The title bar shows the file explorer icon, a redacted path, and the directory 'Desktop/tmp/free\_it\_py\_data\_analysis'. The terminal text shows the execution of 'git status' on the 'master' branch. It reports 'On branch master', 'No commits yet', and 'Untracked files: dummy.py'. A message at the bottom says 'nothing added to commit but untracked files present (use "git add" to track)'.

```
Desktop/tmp/free_it_py_data_analysis
~/Desktop/tmp/free_it_py_data_analysis (master)
$ git status
On branch master

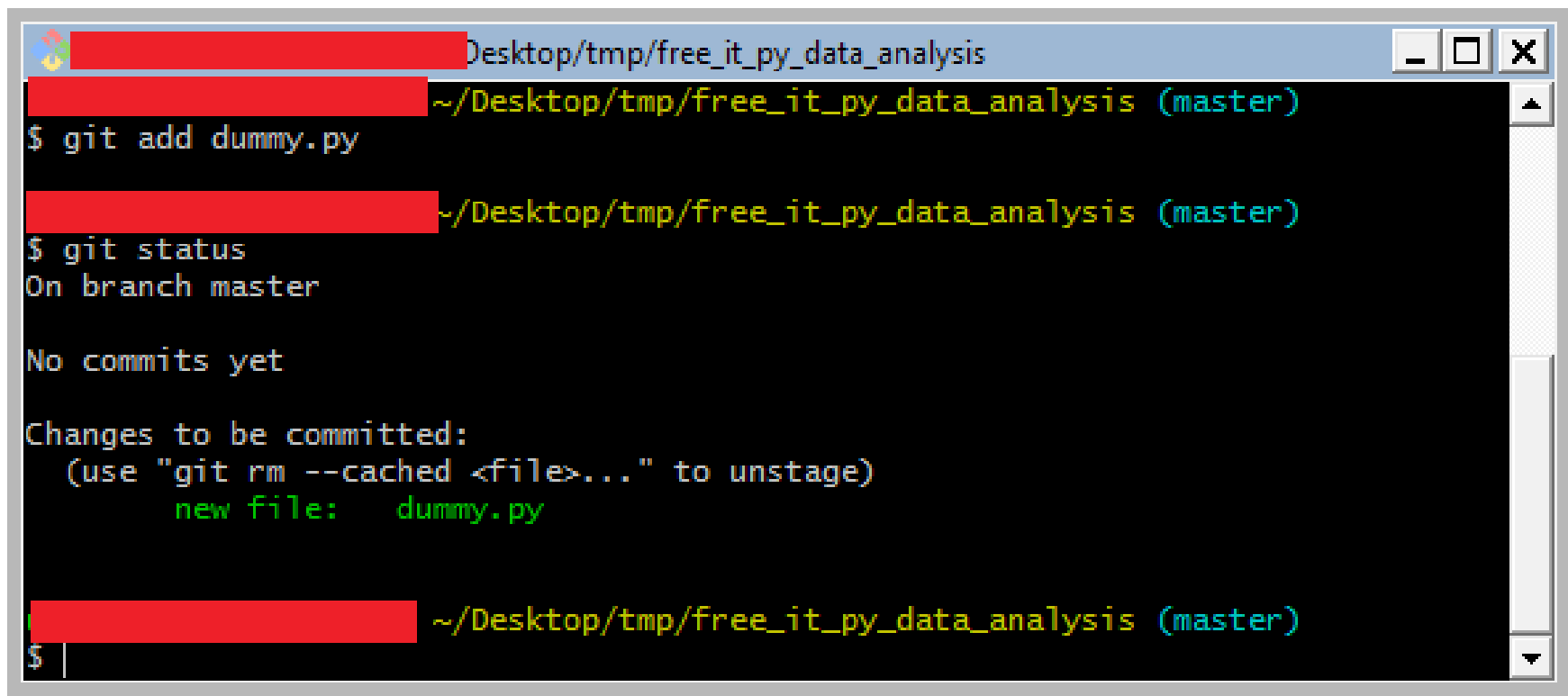
No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dummy.py

nothing added to commit but untracked files present (use "git add" to track)
~/Desktop/tmp/free_it_py_data_analysis (master)
$
```

## СОХРАНЕНИЕ ИЗМЕНЕНИЙ В ОБЛАСТЬ ПОДГОТОВЛЕННЫХ ФАЙЛОВ (STAGING AREA)

**\$ git add dummy.py #чтобы переместить dummy.py в staging area**

A screenshot of a Windows command prompt window. The title bar shows the Windows logo and a redacted path: "Desktop/tmp/free\_it\_py\_data\_analysis". The window contains the following text:

```
~\Desktop/tmp/free_it_py_data_analysis (master)
$ git add dummy.py

~\Desktop/tmp/free_it_py_data_analysis (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   dummy.py

~\Desktop/tmp/free_it_py_data_analysis (master)
$
```

## КОММИТ ПОДГОТОВЛЕННЫХ ФАЙЛОВ

**\$ git commit -m "your message here"**

**\$ git status #ничего нового не осталось в рабочей папке и staging area**

```
~/Desktop/tmp/free_it_py_data_analysis (master)
$ git commit -m "put your message here"
[master (root-commit) 43c3fd1] put your message here
1 file changed, 2 insertions(+)
create mode 100644 dummy.py
```

```
~/Desktop/tmp/free_it_py_data_analysis (master)
$ git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean
```

**\$ git log #увидеть все коммиты**

```
~/Desktop/tmp/free_it_py_data_analysis (master)
$ git log
commit 43c3fd1e25b66fae26312f738ce077973f8956de (HEAD -> master)
Author: <> com>
Date:   Fri Sep 18 16:57:47 2020 +0100

    put your message here
```



## ДОБАВЛЕНИЕ (PUSH) НА СЕРВЕР

**\$ git push #добавить все новые коммиты на сервер**

```
████████████████████ ~/Desktop/tmp/free_it_py_data_analysis (master)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 286 bytes | 95.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/██████████/free_it_py_data_analysis.git
 * [new branch]      master -> master
```

## ДОБАВЛЕНИЕ (PUSH) НА СЕРВЕР

The screenshot shows the GitHub interface for a repository named 'free\_it\_py\_data\_analysis'. The repository is private and has 1 pull request, 0 stars, and 0 forks. The 'Code' tab is selected, showing a commit history. The most recent commit is from 28 minutes ago, with 1 commit in total. The commit message is 'put your message here'. A file named 'dummy.py' is listed in the commit details, also with the message 'put your message here' and timestamp '28 minutes ago'. The right sidebar shows the 'About' section with the text 'No description, website, or topics provided.'

GitHub logo Search or jump to... / Pulls Issues Marketplace Explore

lock icon [redacted] / free\_it\_py\_data\_analysis Private Unwatch 1 Star 0 Fork 0

<> Code ! Issues 🔗 Pull requests 🔄 Actions 📁 Projects 🛡 Security 📈 Insights ...

🔑 master Go to file Add file Code About ⚙

🔑 [redacted] put your message here ... 28 minutes ago 🕒 1

📄 dummy.py put your message here 28 minutes ago

No description, website, or topics provided.

## ТЕРМИНОЛОГИЯ

- **Репозиторий (Repository):** online-версия вашей git-директории
- **Форк (ответвление, Fork):** копия чьего-то репозитория
- **Ветка (бранч, Branch):** версия вашего репозитория
  - По умолчанию, “master” – ваша главная ветка
- **Пулл реквест (Pull Request/Merge Request):** действие по слиянию изменений из одной ветки с другим.
  - Например, когда вы заканчиваете работу над изменениями в одной ветке и хотите обновить вашу основную ветку, добавив изменения в неё.
- **Конфликт (Conflict):** может произойти когда несколько коллег работают в разных ветках.
  - Если коллеги вносят несовместимые изменения в один файл, это ведет к конфликту.
  - Вам не придется столкнуться с этой проблемой в течение курса.