

Федеральное государственное автономное
образовательное учреждение высшего
образования
«Национальный исследовательский
университет
ИТМО»

Факультет Информационных технологий и
программирования

Работа: Изучение системы управления версиями Git

Выполнили:
Миловацкий Никита Владимирович
Худашов Богдан Геннадьевич
М3113

Проверил:

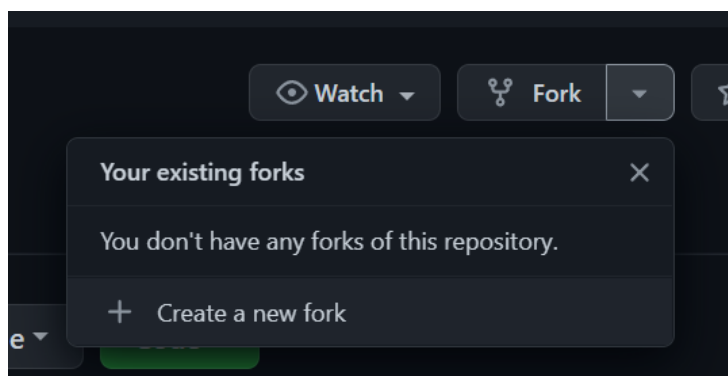
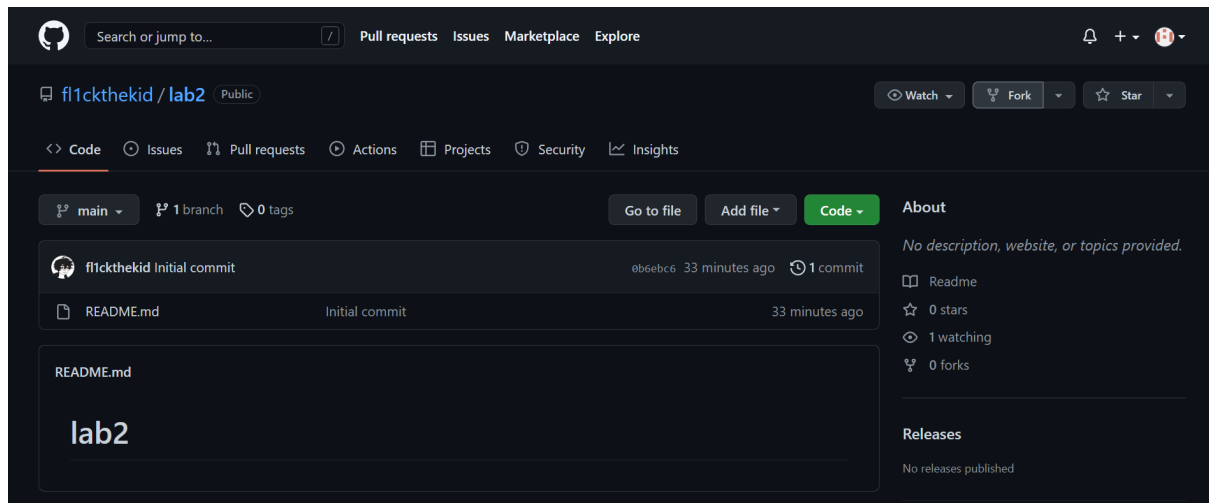
Санкт-Петербург
2022 г.

Задание:

Изучить и описать систему управления версиями Git.

- 1) Первый студент заводит репозиторий, второй делает в нее Pull request.
- 2) Каждому:
 - a. Создание веток по модели GitFlow - https://danielkummer.github.io/git-flow-cheatsheet/index.ru_RU.html - Обязательное наличие веток фич, релиз, девелоп, хотфикс.
 - b. В репозитории обязательно оформлен ReadMe.
 - c. Наличие тегов.
 - d. Submodules
 - e. LFS
- 3) Также всем студентам обязательно подготовить справочник по основным командам Git с примерами. Уметь кратко ответить на вопросы о предназначении основных команд Git.

1. Первый студент заводит репозиторий, второй делает в нее Pull request.



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Owner *



somenicknameidc

Repository name *

/ lab2



By default, forks are named the same as their parent repository. You can customize the name to distinguish it further.

Description (optional)

☒ Copy the `main` branch only

Contribute back to fl1ckthekid/lab2 by adding your own branch. [Learn more.](#)

You are creating a fork in your personal account.

Create fork



somenicknameidc / lab2

Public

forked from fl1ckthekid/lab2

```
C:\Users\nik>cd /d D:\Data
```

```
D:\Data>git clone https://github.com/fl1ckthekid/lab2
```

```
Cloning into 'lab2'...
```

```
remote: Enumerating objects: 3, done.
```

```
remote: Counting objects: 100% (3/3), done.
```

```
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
```

```
Receiving objects: 100% (3/3), done.
```

```
D:\Data>cd lab2
```

```
D:\Data\lab2>_
```

```
D:\Data\lab2>git branch test
```

```
D:\Data\lab2>git checkout test
```

```
Switched to branch 'test'
```

```
D:\Data\lab2>git status
```

```
on branch test
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

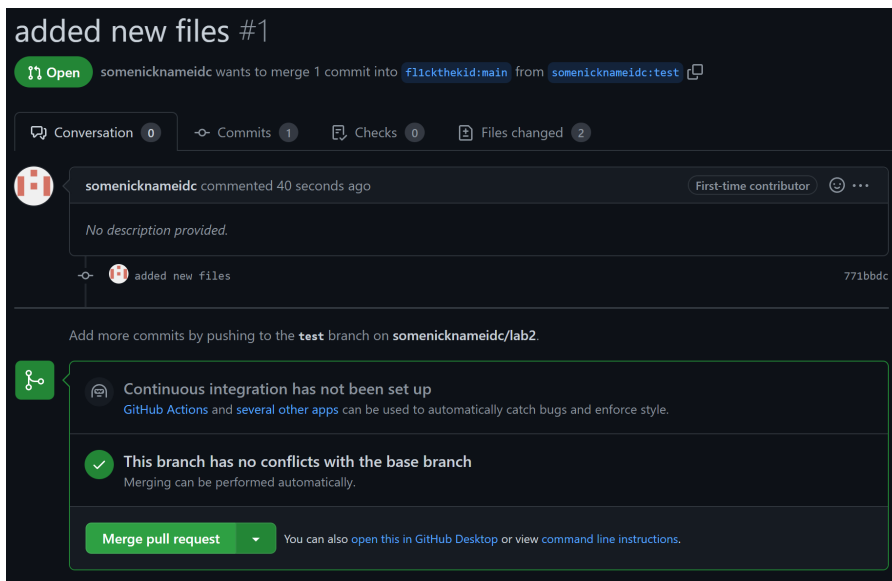
```
circle.py
```

```
square.py
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

```
D:\Data\lab2>git add circle.py
```

```
D:\Data\lab2>git add square.py
```

2. Создание веток по модели GitFlow

- 1) Добавление ветки фичи feature1, объединение с веткой develop, удаление ветки feature1, push ветки в репозиторий.

```

C:\Program Files\Git\lab2>git branch develop

C:\Program Files\Git\lab2>git checkout develop
Switched to branch 'develop'

C:\Program Files\Git\lab2>git branch feature1

C:\Program Files\Git\lab2>git checkout feature1
Switched to branch 'feature1'

C:\Program Files\Git\lab2>git add test1.txt

C:\Program Files\Git\lab2>git status
On branch feature1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   test1.txt

C:\Program Files\Git\lab2>git commit -m "File test1 was added"
[feature1 5fbaa3e] File test1 was added
 1 file changed, 1 insertion(+)
 create mode 100644 test1.txt

C:\Program Files\Git\lab2>git checkout develop
Switched to branch 'develop'

C:\Program Files\Git\lab2>git merge feature1
Updating c983129..5fbaa3e
Fast-forward
 test1.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 test1.txt

C:\Program Files\Git\lab2>git branch -d feature1
Deleted branch feature1 (was 5fbaa3e).

C:\Program Files\Git\lab2>git push origin develop
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 282 bytes | 282.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/fl1ckthekid/lab2/pull/new/develop
remote:
To https://github.com/fl1ckthekid/lab2
 * [new branch]      develop -> develop

```

- 2) Добавление ветки релиза release1, объединение с веткой main и develop, удаление ветки release1, создание тега v1.0, push в репозиторий.

```
C:\Program Files\Git\lab2>git branch release1

C:\Program Files\Git\lab2>git checkout release1
Switched to branch 'release1'

C:\Program Files\Git\lab2>git add main.c
warning: in the working copy of 'main.c', LF will be replaced by CRLF the next time Git touches it

C:\Program Files\Git\lab2>git status
On branch release1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   main.c

C:\Program Files\Git\lab2>git coommit -m "main.c was added"
git: 'coomit' is not a git command. See 'git --help'.

The most similar command is
    commit

C:\Program Files\Git\lab2>git commit -m "main.c was added"
[release1 0fd01b2] main.c was added
 1 file changed, 22 insertions(+)
 create mode 100644 main.c

C:\Program Files\Git\lab2>git branch
  develop
  main
* release1

C:\Program Files\Git\lab2>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Program Files\Git\lab2>git merge release1
Updating c983129..0fd01b2
Fast-forward
 main.c      | 22 ++++++
 test1.txt   |  1 +
 2 files changed, 23 insertions(+)
 create mode 100644 main.c
 create mode 100644 test1.txt

C:\Program Files\Git\lab2>git tag v1.0
```

```
C:\Program Files\Git\lab2>git checkout develop
Switched to branch 'develop'

C:\Program Files\Git\lab2>git merge release1
Updating 5fbaa3e..0fd01b2
Fast-forward
 main.c | 22 +++++++++++++++++++++
 1 file changed, 22 insertions(+)
 create mode 100644 main.c

C:\Program Files\Git\lab2>git branch -d release1
Deleted branch release1 (was 0fd01b2).

C:\Program Files\Git\lab2>git push --tags
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 545 bytes | 545.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/fl1ckthekid/lab2
 * [new tag]          v1.0 -> v1.0
```

```
C:\Program Files\Git\lab2>git pull origin
Merge made by the 'ort' strategy.

C:\Program Files\Git\lab2>git push origin main
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 248 bytes | 248.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/fl1ckthekid/lab2
 bedfb70..858a2ff  main -> main
```

- 3) Добавление хотфикс ветки hotfix1, объединение с веткой main и develop, создание тега fix1, удаление ветки hotfix1, push в репозиторий.


```
D:\Data\lab2>git branch hotfix1

D:\Data\lab2>git checkout hotfix1
Switched to branch 'hotfix1'

D:\Data\lab2>git add 1F.cpp

D:\Data\lab2>git commit -m "file 1F.cpp was added"
[hotfix1 28eacbe] file 1F.cpp was added
 1 file changed, 62 insertions(+)
 create mode 100644 1F.cpp
```

```
D:\Data\lab2>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 6 commits.
  (use "git push" to publish your local commits)

D:\Data\lab2>git merge hotfix1
Updating 858a2ff..28eacbe
Fast-forward
 1F.cpp | 62 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 62 insertions(+)
 create mode 100644 1F.cpp
```

```

D:\Data\lab2>git tag fix1

D:\Data\lab2>git checkout develop
Switched to branch 'develop'

D:\Data\lab2>git merge hotfix1
Updating 858a2ff..28eacbe
Fast-forward
 1F.cpp | 62 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 62 insertions(+)
 create mode 100644 1F.cpp

D:\Data\lab2>git branch -d hotfix1
Deleted branch hotfix1 (was 28eacbe).

D:\Data\lab2>git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 1010 bytes | 1010.00 KiB/s, done.
Total 3 (delta 1), reused 2 (delta 1), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/somenicknameidc/lab2
   0b6ebc6..28eacbe  main -> main


```

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 repository: fl1ckthekid/lab2
base: main
head repository: somenicknameidc/lab2
compare: main

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



file 1F.cpp was added

Write

Preview

H B I ≡ <> 🔗 ≡ ≡ ≡ @ ↗ ↶

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

📎

☒ Allow edits by maintainers ⓘ

Create pull request

▼

4) Добавление подмодуля герo3113 в репозиторий lab2.

```

C:\Program Files\Git\lab2>git submodule add https://github.com/fl1ckthekid/repo3113
Cloning into 'C:/Program Files/Git/lab2/repo3113'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
warning: in the working copy of '.gitmodules', LF will be replaced by CRLF the next time Git touches it

C:\Program Files\Git\lab2>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   .gitmodules
    new file:   repo3113

C:\Program Files\Git\lab2>git commit -m "submodule repo3113 was added"
[main 003db72] submodule repo3113 was added
 2 files changed, 4 insertions(+)
 create mode 100644 .gitmodules
 create mode 160000 repo3113

```

- 5) Инициализация LFS, отслеживание файла 8k_picture.jpg с использованием LFS.

```

C:\Program Files\Git\lab2>git lfs install
Updated Git hooks.
Git LFS initialized.

```

```

C:\Program Files\Git\lab2>git add 8k_picture.jpg

C:\Program Files\Git\lab2>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   8k_picture.jpg

C:\Program Files\Git\lab2>git commit -m "8k_picture was added"
[main 3a96a9d] 8k_picture was added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 8k_picture.jpg

C:\Program Files\Git\lab2>git lfs track ""8k_picture"
Tracking "8k_picture"

C:\Program Files\Git\lab2>git add .gitattributes

C:\Program Files\Git\lab2>git commit -m "Tracked 8k_picture file with Git LFS"
[main b7e5e20] Tracked 8k_picture file with Git LFS
 1 file changed, 1 insertion(+)
 create mode 100644 .gitattributes

```

- 6) Оформление файла README.md репозитория.

<> Edit file

Preview

```
1 # lab2
2 ## Лабораторная работа 2: Изучение системы управления версиями Git
3 ### Задание
4 Нужно изучить и описать систему управления версиями Git в объёме не
5 более пяти страниц, не считая титульника. Формат сдачи следующий: я
6 задаю вопросы по основным командам Git, вам нужно дать ответы, опираясь
7 на ваши отчеты.
8 За каждый неправильный ответ отнимается один балл от числа максимального
9 количества баллов за работу.
10
11 ##### Разбиваемся по парам и создаем
12 1. Первый студент заводит репозиторий, второй делает в нее Pull request.
13 2. Каждому:
14     - Создание веток по модели GitFlow - https://danielkummer.github.io/git-flow-cheatsheet/index.ru\_RU.html
15     - Обязательное наличие веток фич, релиз, девелоп, хотфикс.
16     - В репозитории обязательно оформлен ReadMe.
17     - Наличие тегов.
18     - Submodules
19     - LFS
20 3. Также всем студентам обязательно подготовить справочник по основным
21 командам Git с примерами. Уметь кратко ответить на вопросы о предназначении
22 основных команд Git.
23
24 ### Literature:
25 - [Книга про Git](https://git-scm.com/book/ru/v1)
26 - [Как использовать Git](https://www.atlassian.com/ru/git)
27
28 ### GitHub accounts
29 - [Богдан](https://github.com/flickthekid)
```

☰ README.md

lab2

Лабораторная работа 2: Изучение системы управления версиями Git

Задание

Нужно изучить и описать систему управления версиями Git в объёме не более пяти страниц, не считая титульника. Формат сдачи следующий: я задаю вопросы по основным командам Git, вам нужно дать ответы, опираясь на ваши отчеты. За каждый неправильный ответ отнимается один балл от числа максимального количества баллов за работу.

Разбиваемся по парам и создаем

1. Первый студент заводит репозиторий, второй делает в нее Pull request.
2. Каждому:
 - Создание веток по модели GitFlow - https://danielkummer.github.io/git-flow-cheatsheet/index.ru_RU.html
 - Обязательное наличие веток фич, релиз, девелоп, хотфикс.

- В репозитории обязательно оформлен ReadMe.
- Наличие тегов.
- Submodules
- LFS

3. Также всем студентам обязательно подготовить справочник по основным командам Git с примерами.
Уметь кратко ответить на вопросы о предназначении основных команд Git.

Literature:

- [Книга про Git](#)
- [Как использовать Git](#)

GitHub accounts

- [Богдан](#)
- [Никита](#)

Branch links

- [develop](#)
- [main](#)

Tag links

- [v1.0](#)
- [fix1](#)

Results

- [Отчёт](#)
- [Справочник](#)