

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №1.1

Исследование
основных возможностей Git и GitHub

Выполнил студент группы

ИВТ-б-о-21-1

Мальцев Н.А. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.

(подпись)

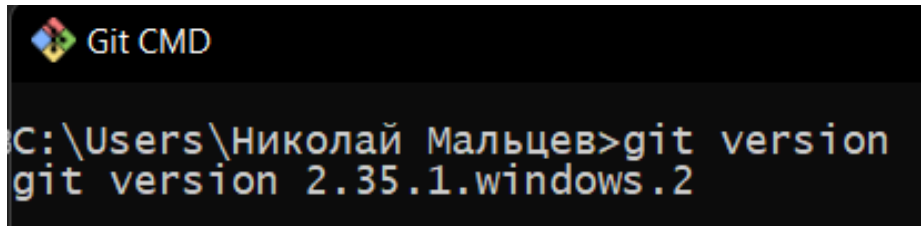
Ставрополь 2022

Исследование основных возможностей Git и GitHub.

Цель работы: исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга GitHub.

Порядок выполнения работы:

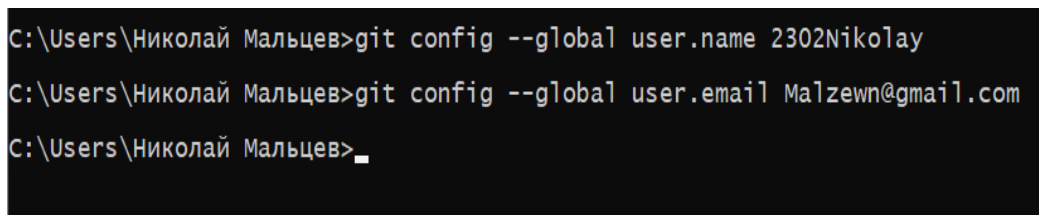
- 1) Произвел установку Git на компьютер.
- 2) При помощи командной строки удостоверился в корректности установки:



```
Git CMD
C:\Users\Николай Мальцев>git version
git version 2.35.1.windows.2
```

Рисунок 1. Результат команды git version

- 3) В настройки Git добавлены имя и адрес электронной почты, которая связана с учетной записью GitHub.



```
C:\Users\Николай Мальцев>git config --global user.name 2302Nikolay
C:\Users\Николай Мальцев>git config --global user.email Malzewn@gmail.com
C:\Users\Николай Мальцев>_
```

Рисунок 2. Добавление имени и электронной почты

- 4) Создан первый репозиторий на GitHub:

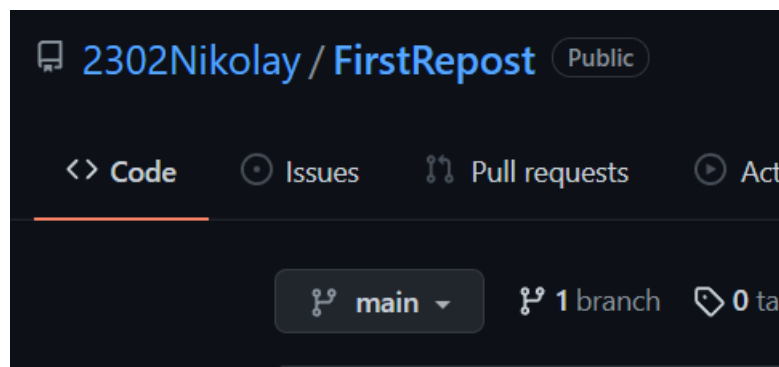


Рисунок 3. Созданный на github репозиторий

- 5) Создан проект консольного приложения на языке C++ в VisualStudio, написана небольшая программа.
- 6) В папке проекта инициализируем репозиторий:

```
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git init
Initialized empty Git repository in C:/Users/Николай Мальцев/OneDrive/Рабочий стол/Test1/Test1/.git/
```

Рисунок 4. Инициализация репозитория

7) Добавление проекта в локальный репозиторий при помощи команды `git add`. и проверка корректности его добавления при помощи команды `git status`:

```
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git add .
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .vs/Test1/project-colors.json
    new file:   .vs/Test1/v17/.suo
    new file:   .vs/Test1/v17/Browse.VC.db
    new file:   .vs/Test1/v17/ipch/AutoPCH/5bb5f12f1f1adf63/TEST1.ipch
    new file:   Test1.sln
    new file:   Test1/Test1.cpp
    new file:   Test1/Test1.vcxproj
    new file:   Test1/Test1.vcxproj.filters
    new file:   Test1/Test1.vcxproj.user
```

Рисунок 5. Результат выполнения команд

8) Делает первый коммит:

```
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git commit -m"First commit"
[master (root-commit) ffa4af8] First commit
24 files changed, 239 insertions(+)
create mode 100644 .vs/Test1/project-colors.json
create mode 100644 .vs/Test1/v17/.suo
create mode 100644 .vs/Test1/v17/Browse.VC.db
create mode 100644 .vs/Test1/v17/ipch/AutoPCH/5bb5f12f1f1adf63/TEST1.ipch
create mode 100644 Test1.sln
create mode 100644 Test1/Test1.cpp
create mode 100644 Test1/Test1.vcxproj
create mode 100644 Test1/Test1.vcxproj.filters
create mode 100644 Test1/Test1.vcxproj.user
create mode 100644 Test1/x64/Debug/Test1.exe.recipe
create mode 100644 Test1/x64/Debug/Test1.ilkg
create mode 100644 Test1/x64/Debug/Test1.log
create mode 100644 Test1/x64/Debug/Test1.obj
create mode 100644 Test1/x64/Debug/Test1.tlog/CL.command.1.tlog
create mode 100644 Test1/x64/Debug/Test1.tlog/CL.read.1.tlog
create mode 100644 Test1/x64/Debug/Test1.tlog/CL.write.1.tlog
create mode 100644 Test1/x64/Debug/Test1.tlog/Test1.lastbuildstate
create mode 100644 Test1/x64/Debug/Test1.tlog/link.command.1.tlog
create mode 100644 Test1/x64/Debug/Test1.tlog/link.read.1.tlog
create mode 100644 Test1/x64/Debug/Test1.tlog/link.write.1.tlog
create mode 100644 Test1/x64/Debug/vc143.idb
create mode 100644 Test1/x64/Debug/vc143.pdb
```

Рисунок 6. Первый коммит

9) Сделаны некоторые изменения в проекте и проверяем наличие этих изменений с помощью команды `git status`:

```

PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .vs/Test1/v17/.suo
        modified:   .vs/Test1/v17/Browse.VC.db
        modified:   Test1/Test1.cpp
        modified:   Test1/x64/Debug/Test1.ilc
        modified:   Test1/x64/Debug/Test1.obj
        modified:   Test1/x64/Debug/Test1.tlog/CL.read.1.tlog
        modified:   Test1/x64/Debug/Test1.tlog/link.read.1.tlog
        modified:   Test1/x64/Debug/Test1.tlog/link.write.1.tlog
        modified:   Test1/x64/Debug/vc143.idb
        modified:   Test1/x64/Debug/vc143.pdb
        modified:   x64/Debug/Test1.exe
        modified:   x64/Debug/Test1.pdb

```

Рисунок 7. Изменения в проекте

10) Производится добавление файлов в локальный репозиторий, делается второй коммит, проверяется корректность произведенных операций при помощи git status:

```

PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git add .
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git commit -m"Second commit"
[master d6ca3a9] Second commit
12 files changed, 6 insertions(+), 1 deletion(-)
rewrite .vs/Test1/v17/.suo (68%)
rewrite Test1/x64/Debug/Test1.obj (79%)
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git status
On branch master
nothing to commit, working tree clean

```

Рисунок 8. Результат выполнения команд

11) Производится еще несколько изменений в исходном коде. Каждое изменение фиксируется коммитами.

12) Локальный репозиторий добавляется в ранее созданный удаленный на GitHub:

```

PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git remote add origin https://github.com/2302Nikolay/FirstRepost.git
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git push -u origin master
Enumerating objects: 80, done.
Counting objects: 100% (80/80), done.
Delta compression using up to 12 threads
Compressing objects: 100% (68/68), done.
Writing objects: 100% (80/80), 12.39 MiB | 2.40 MiB/s, done.
Total 80 (delta 26), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (26/26), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/2302Nikolay/FirstRepost/pull/new/master
remote:
To https://github.com/2302Nikolay/FirstRepost.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

```

Рисунок 9. Добавление в удаленный репозиторий

13) В файл README добавлены название группы и ФИО:

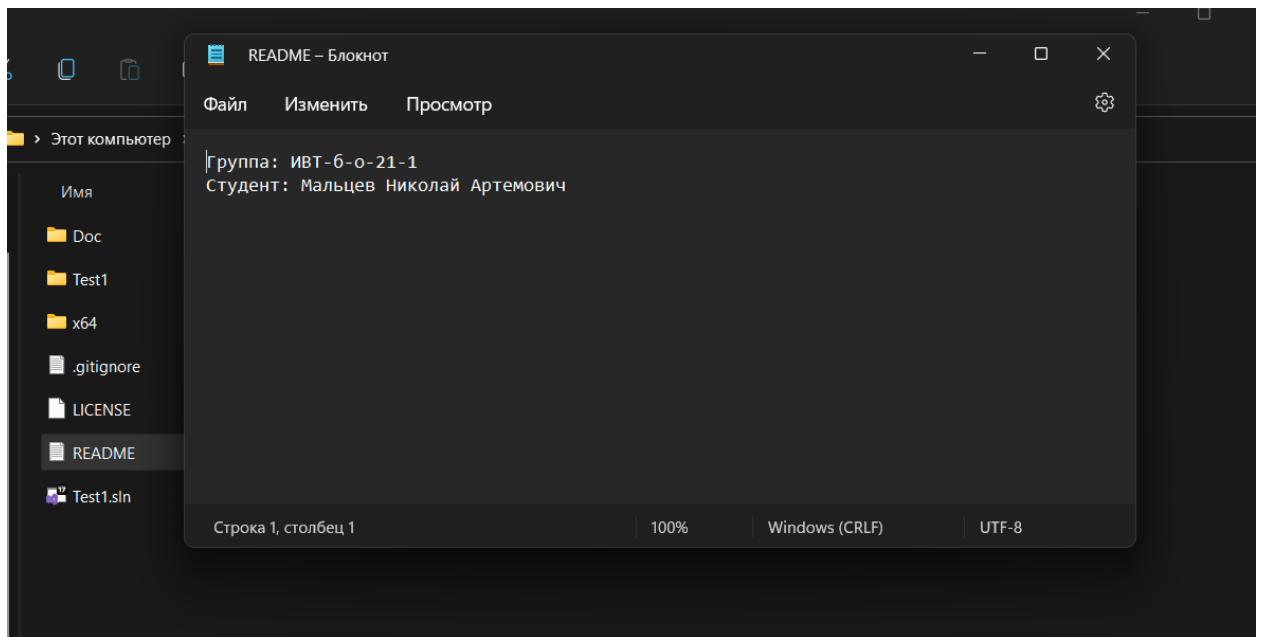


Рисунок 10. Изменение файла README

14) Сделан коммит:

```
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git add .
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git commit -m"Добавил группу и фио"
[main 8803221] Добавил группу и фио
1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\Николай Мальцев\OneDrive\Рабочий стол\Test1\Test1> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Рисунок 11. Добавление файла README

Контрольные вопросы:

1) Что такое СКВ и каково её назначение?

СКВ или Система контроля версий предназначена для контроля изменений в файлах с целью дальнейшей возможности вернуться к определенным старым версиям этих файлов.

2) К какой СКВ относится Git?

Git относится к распределенным системам, поэтому не зависит от центрального сервера, где хранятся файлы.

3) В чем концептуальное отличие Git от других СКВ?

Обычно СКВ хранят информацию в виде списка изменений в файле. Git же хранит информацию в виде набора файлов и изменений, сделанных в каждом файле.

4) Как обеспечивается целостность хранимых данных в Git?

В Git для всего вычисляется хеш-сумма, и потом происходит сохранение. В дальнейшем обращение к сохранённым объектам происходит по этой хеш-сумме. Это значит, что невозможно изменить содержимое файла или директории так, чтобы Git не узнал об этом.

5) В каких состояниях могут находиться файлы в Git? Как связаны эти состояния?

Зафиксированный файл – файл сохранён в локальном репозитории.

Измененный файл – файл, который был изменен, но ещё не был зафиксирован.

Подготовленный файл — это изменённый файл, отмеченный для включения в следующий коммит.

6) Профиль – ваша публичная страница на GitHub, как и в социальных сетях. Когда мы ищем работу в качестве программиста, работодатели могут посмотреть наш профиль GitHub и принять его во внимание, когда будут решать, брать нас на работу или нет.

7) Локальный – находящийся непосредственно на компьютере пользователя, и удаленный – находящийся на удаленном сервере.

8) Основными этапами являются «Регистрация, создание репозитория, клонирование репозитория».

9) Убеждаемся, что Git установлен используя команду: `git version`; связываем локальный репозиторий с удаленным; делаем коммиты и отправляем изменения в удаленный репозиторий.

10) В правом верхнем углу, рядом с аватаром нажимается кнопка «+», производится переход к созданию нового репозитория. Здесь вводится имя репозитория, тип репозитория (закрытый или публичный), добавляется файл `gitignore` для нужного языка программирования и среды разработки, выбирается лицензия.

11) Microsoft Reciprocal License, The Code Project Open License (CPO), The Common Development and Distribution License (CDDL), The Microsoft Public License (Ms-PL), The Mozilla Public License 1.1 (MPL 1.1), The Common Public License Version 1.0 (CPL), The Eclipse Public License 1.0, The MIT License, The BSD License, The Apache License, Version 2.0, The Creative Commons Attribution-ShareAlike 2.5 License, The zlib/libpng License, A Public Domain dedication, The Creative Commons Attribution 3.0 Unported License, The Creative Commons Attribution-Share Alike 3.0 Unported License, The Creative Commons Attribution-NoDerivatives 3.0 Unported, The GNU Lesser General Public License (LGPLv3), The GNU General Public License (GPLv3).

12) После создания репозитория его необходимо клонировать на компьютер. Для этого на странице репозитория необходимо найти адрес репозитория для клонирования. После его копирования в командной строке вводится команда `git clone 'ссылка на репозиторий'`.

13) Используя команду `git status`.

14) Файлы на удаленном репозитории будут обновлены.

15) Клонировать репозиторий на каждый из компьютеров, используя команду `git clone` и ссылку, используем команду `git pull` для синхронизации.

16) GitLab, SourceForge, Launchpad.

17) GitLab — альтернатива GitHub. GitLab предоставляет не только веб-сервис для совместной работы, но и программное обеспечение с открытым исходным кодом

Launchpad — платформа для совместной работы над программным обеспечением разработчика Ubuntu. На ней размещены PPA-репозитории Ubuntu, откуда пользователи загружают приложения и обновления.

Вывод: в ходе работы были исследованы базовые возможности системы контроля версий Git и веб-сервиса для хостинга GitHub.