

**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 №2

Исследование
основных возможностей Git и GitHub

Выполнил студент группы

ИВТ-б-о-21-1

Урусов М.А. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.

(подпись)

Ставрополь 2022

Тема: исследование возможностей Git для работы с локальными репозиториями.

Цель работы: исследовать базовые возможности системы контроля версий Git для работы с локальными репозиториями.

Выполнение работы.

```
D:\git\Laba_2>git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Рисунок 1. Клонирование репозитория

```
C:\Users\den-n>cd Laba2
C:\Users\den-n\Laba2>git log
commit 38cc6ced78fd4339996b0cff65c147725194dfdd (HEAD -> main, origin/main, origin/HEAD)
Author: MaksimUrusov <den-ney@bk.ru>
Date:   Wed May 11 00:52:39 2022 +0300

    test1

commit 9b95931cf1afb4102ba3ca4f98810f6f37c549d7
Author: MaksimUrusov <99472213+MaksimUrusov@users.noreply.github.com>
Date:   Wed May 11 00:43:47 2022 +0300

    Initial commit
C:\Users\den-n\Laba2>
```

Рисунок 2. Коммиты

```

C:\Users\den-n\Laba2>git log -p -2
commit 38cc6ced78fd4339996b0cfff65c147725194dfdd (HEAD -> main, origin/main, origin/HEAD)
Author: MaksimUrusov <den-ney@bk.ru>
Date: Wed May 11 00:52:39 2022 +0300

    test1

diff --git a/.gitignore b/.gitignore
index b6e4761..2a658e0 100644
--- a/.gitignore
+++ b/.gitignore
@@ -127,3 +127,509 @@ dmy.py.json

# Pyre type checker
.pyre/

# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]

```

Рисунок 3. Вывод коммитов с определённым условием

```

C:\Users\den-n\Laba2>git log --stat
commit 38cc6ced78fd4339996b0cfff65c147725194dfdd (HEAD -> main, origin/main, origin/HEAD)
Author: MaksimUrusov <den-ney@bk.ru>
Date: Wed May 11 00:52:39 2022 +0300

    test1

.gitignore | 506 ++++++
LICENSE | 2 +-
Laba_2/Laba_2.sln | 31 ++
Laba_2/Laba_2/Laba_2.cpp | 17 +
Laba_2/Laba_2/Laba_2.vcxproj | 147 ++++++
Laba_2/Laba_2/Laba_2.vcxproj.filters | 22 ++
README.md | 5 +-
"doc\320\233\320\260\320\261\320\260 2.pdf" | Bin 0 -> 439205 bytes
8 files changed, 728 insertions(+), 2 deletions(-)

commit 9b95931cf1afb4102ba3ca4f98810f6f37c549d7
Author: MaksimUrusov <99472213+MaksimUrusov@users.noreply.github.com>
Date: Wed May 11 00:43:47 2022 +0300

    Initial commit

.gitignore | 129 ++++++
LICENSE | 21 ++++++
README.md | 1 +
3 files changed, 151 insertions(+)
C:\Users\den-n\Laba2>

```

Рисунок 4. Сокращенная статистика

```
D:\git\Laba_2>git log --pretty=oneline
c0e45a4b08bd6d5074ff171b3edffb2bac400798 (HEAD -> main, origin/main, origin/HEAD) Небольшое изменение
895eb8a4b03716bd17ece3cc34d59fedd1a3e92b Добавление файлов для работы Python
8170d4371ad486ad9ca8e7c88d5fc89c5be6282d Имя и группа
3fc1b9c25fd13efb0bf0a34f2801990494ad2303 Initial commit

D:\git\Laba_2>
```

Рисунок 5. Коммиты

```
C:\Users\den-n\Laba2>git log --pretty=format:"%h - %an, %ar : %s"
Fromat:38cc6ce - MaksimUrusov, 9 minutes ago : test1
Fromat:9b95931 - MaksimUrusov, 18 minutes ago : Initial commit

C:\Users\den-n\Laba2>
```

Рисунок 6. Коммиты с определённым форматом

```
D:\git\Laba_2>git log --pretty=format:"%h %s" --graph
* c0e45a4 Небольшое изменение
* 895eb8a Добавление файлов для работы Python
* 8170d43 Имя и группа
* 3fc1b9c Initial commit

D:\git\Laba_2>
```

Рисунок 7. Текущая ветка и история слияний

The screenshot shows the GitHub interface for the repository 'MaksimUrusov/Laba2'. The 'main' branch is selected, showing a commit history table with columns for commit hash, message, and time. The commit messages are: 'test1' (38cc6ce, 28 minutes ago), 'test1' (895eb8a, 28 minutes ago), 'test1' (8170d43, 28 minutes ago), 'test1' (3fc1b9c, 28 minutes ago), and 'Initial commit' (38cc6ce, 28 minutes ago). The repository files section shows 'Laba_2' (Urusov Maksim IBT-6-a-21-1), '.gitignore', 'LICENSE', and 'README.md'. The right sidebar contains 'About' (No description, website, or topics provided), 'Releases' (No releases published), and 'Packages' (No packages published).

Рисунок 8. Репозиторий

```
#include <iostream>

using namespace std;

int main()
{
    int a, b, c;
    cout << "Enter values - \n";
    cin >> a, b, c;
    if (a > 0) {
        cout << b + c;
    }
    else {
        cout << b * c;
    }
    return 0;
}
```

Рисунок 9. Код

```

D:\git\Laba_2> git log --graph --pretty=oneline --abbrev-commit
* 25c8506 (HEAD -> main) 3
* 2656d02 3
* 629b8c4 2
* 468ef76 2
* 3c93a22 2
* 0d5d5b0 1
* f15dc85 1
* 4985161 1
* 7b25b73 (origin/main, origin/HEAD) создание проекта
* 204f926 перешёл с python на c++
* c0e45a4 Небольшое изменение
* 895eb8a Добавление файлов для работы Python
* 8170d43 Имя и группа
* 3fc1b9c Initial commit

```

Рисунок 10. Просмотрел историю хранилища

```

Author: MaksimUrusov <den-ney@bk.ru>
Date: Sun May 8 12:06:14 2022 +0300

    test2

diff --git a/Laba2/main.py b/Laba2/main.py
new file mode 100644
index 0000000..1123c2e
--- /dev/null
+++ b/Laba2/main.py
@@ -0,0 +1,18 @@
+num1 = float(input('Enter your first number: '))
+Operator = input('Enter operator: ')
+num2 = float(input('Enter your second number: '))
+
+if Operator == '+':
+    print(num1 + num2)
+elif Operator == '-':
+    print(num1 - num2)
+elif Operator == '/':
+    print(num1 / num2)
+elif Operator == '*':
+    print(num1 * num2)
+
+else:
+    print('Not a valid operator')
(END)

```

Рисунок 11. Просмотрел коммиты с помощью git show

```
D:\git\Laba_2\Laba_2\Laba_2>git status
On branch main
Your branch is ahead of 'origin/main' by 8 commits.
  (use "git push" to publish your local commits)

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:   Laba_2.cpp

no changes added to commit (use "git add" and/or "git commit -a")

D:\git\Laba_2\Laba_2\Laba_2>git add .

D:\git\Laba_2\Laba_2\Laba_2>git commit -m "Удаление файла"
[main 8ace796] Удаление файла
 1 file changed, 1 insertion(+), 17 deletions(-)
```

Рисунок 12. Удалённый код

```
D:\git\Laba_2\Laba_2\Laba_2>git reset --hard HEAD~1
HEAD is now at 25c8506 3

D:\git\Laba_2\Laba_2\Laba_2>
```

Рисунок 13. Откат версии

```

1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int a, b, c;
8      cout << "Enter values - \n";
9      cin >> a, b, c;
10     if (a > 0) {
11         cout << b + c;
12     }
13     else {
14         cout << b * c;
15     }
16     return 0;
17 }
18

```

Рисунок 14. Восстановленный код

Вывод: команда `git -checkout <FileName>` удаляет изменения произошедшие с файлом в репозитории до коммита.

1. Как выполнить историю коммитов в Git? Какие существуют дополнительные опции для просмотра истории коммитов?

Наиболее простой и в то же время мощный инструмент для этого — команда `git log`. По умолчанию, без аргументов, `git log` выводит список коммитов созданных в данном репозитории в обратном хронологическом порядке. То есть самые последние коммиты показываются первыми. Одна из опций, когда вы хотите увидеть сокращенную статистику для каждого коммита, вы можете использовать опцию `-stat`. Вторая опция (одна из самых полезных аргументов) является `-p` или `-- patch`, который показывает разницу (выводит патч), внесенную в каждый коммит. Так же вы можете ограничить количество записей в выводе команды; используйте параметр `-2` для вывода только двух записей (пример команды `git log -p -2`). Третья действительно полезная опция это `--pretty`. Она меняет формат вывода. Существует несколько встроенных вариантов отображения. Опция `oneline` выводит каждый коммит в одну строку,

что может быть очень удобным если вы просматриваете большое количество коммитов. К тому же, опции `short`, `full` и `fuller` делают вывод приблизительно в том же формате, но с меньшим или большим количеством информации соответственно. Наиболее интересной опцией является `format`, которая позволяет указать формат для вывода информации. Особенно это может быть полезным, когда вы хотите сгенерировать вывод для автоматического анализа — так как вы указываете формат явно, он не будет изменен даже после обновления Git. Для опции `git log --pretty=format` существуют различного рода опции для изменения формата отображения.

2. Как ограничить вывод при просмотре истории коммитов?

Для ограничения может использоваться функция `git log`, где `n` — число записей. Также, существуют опции для ограничения вывода по времени, такие как `--since` и `--until`, они являются очень удобными. Например, следующая команда покажет список коммитов, сделанных за последние две недели: `git log --since=2.weeks`. Эта команда работает с большим количеством форматов — вы можете указать определенную дату вида `2008-01-15` или же относительную дату, например `2 years 1 day 3 minutes ago`. Также вы можете фильтровать список коммитов по заданным параметрам. Опция `--author` дает возможность фильтровать по автору коммита, а опция `--grep` (показывает только коммиты, сообщение которых содержит указанную строку) искать по ключевым словам в сообщении коммита.⁸ Функция `-S` показывает только коммиты, в которых изменение в коде повлекло за собой добавление или удаление указанной строки.

3. Как внести изменения в уже сделанный коммит?

Внести изменения можно с помощью команды `git commit --amend`. Эта команда берёт индекс и применяет его к последнему

коммиту. Если после последнего коммита не было никаких проиндексированных изменений (например, вы запустили приведённую команду сразу после предыдущего коммита), то состояние проекта будет абсолютно таким же и всё, что мы изменим, это комментарий к коммиту. Для того, чтобы внести необходимые изменения - нам нужно проиндексировать их и выполнить команду `git commit --amend`. `git commit -m 'initial commit'` `git add forgotten_file` `git commit --amend` Эффект от выполнения этой команды такой, как будто мы не выполнили предыдущий коммит, а еще раз выполнили команду `git add` и выполнили коммит.

4. Как отменить индексацию файла в Git?

Например, вы изменили два файла и хотите добавить их в разные коммиты, но случайно выполнили команду `git add *` и добавили в индекс оба. Как исключить из индекса один из них? Команда `git status` напомним вам: Прямо под текстом «Changes to be committed» говорится: используйте `git reset HEAD` для исключения из индекса.

5. Как отменить изменения в файле?

С помощью команды `git checkout --`.

6. Что такое удаленный репозиторий Git?

Удалённый репозиторий это своего рода наше облако, в которое мы сохраняем те или иные изменения в нашей программе/коде/файлах.

7. Как выполнить просмотр удаленных репозитория данного локального репозитория?

Для того, чтобы просмотреть список настроенных удалённых репозитория, необходимо запустить команду `git remote`. Также можно указать ключ `-v`, чтобы просмотреть адреса для чтения и записи, привязанные к репозиторию. Пример: `git remote -v`

8. Как добавить удаленный репозиторий для данного локального репозитория?

Для того, чтобы добавить удалённый репозиторий и присвоить ему имя (shortname), просто выполните команду `git remote add` .

9. Как выполнить отправку/получение изменений с удаленного репозитория?

Если необходимо получить изменения, которые есть у Пола, но нету у вас, вы можете выполнить команду `git fetch` . Важно отметить, что команда `git fetch` забирает данные в ваш локальный репозиторий, но не сливает их с какими-либо вашими наработками и не модифицирует то, над чем вы работаете в данный момент. Вам необходимо вручную слить эти данные с вашими, когда вы будете готовы. Если ветка настроена на отслеживание удалённой ветки, то вы можете использовать команду `git pull` чтобы автоматически получить изменения из удалённой ветки и слить их со своей текущей. Выполнение `git pull`, как правило, извлекает (fetch) данные с сервера, с которого вы изначально10 клонировали, и автоматически пытается слить (merge) их с кодом, над которым вы в данный момент работаете. Чтобы отправить изменения на удалённый репозиторий необходимо отправить их в удалённый репозиторий. Команда для этого действия простая: `git push` .

10. Как выполнить просмотр удаленного репозитория?

Для просмотра удалённого репозитория, можно использовать команду `git remote show` .

11. Каково назначение тэгов Git?

Теги - это ссылки указывающие на определённые версии кода/написанной программы. Они удобно чтобы в случае чего вернуться к нужному моменту. Также при помощи тегов можно помечать важные моменты.

12. Как осуществляется работа с тэгами Git?

Просмотреть наличие тегов можно с помощью команды: `git tag`. А назначить (указать, добавить тег) можно с помощью команды `git tag -a v1.4(версия изначальная) -m "Название"`. С помощью команды `git show` вы можете посмотреть данные тега вместе с коммитом: `git show v1.4`. Отправка тегов, по умолчанию, команда `git push` не отправляет теги на удалённые сервера. После создания теги нужно отправлять явно на удалённый сервер. Процесс аналогичен отправке веток — достаточно выполнить команду `git push origin`. Для отправки всех тегов можно использовать команду `git push origin tags`. Для удаления тега в локальном репозитории достаточно выполнить команду `git tag -d`. Например, удалить созданный ранее легковесный тег можно следующим образом: `git tag -d v1.4-lw`. Для удаления тега из внешнего репозитория используется команда `git push origin --delete`. Если вы хотите получить версии файлов, на которые указывает тег, то вы можете сделать `git checkout` для тега пример: `git checkout -b version2 v2.0.0`.

13. Самостоятельно изучите назначение флага `--prune` в командах `git fetch` и `git push`. Каково назначение этого флага?

`Git fetch --prune` команда получения всех изменений с репозитория GitHub. В команде `git push --prune` удаляет удаленные ветки, у которых нет локального аналога. Вывод: исследовал базовые возможности системы контроля версий `git` для работы с локальными репозиториями. Также, благодаря созданию тегов и пункту 7 лабораторной работы после изменения файлов освоил возможность отката к заданной версии.

Вывод: исследовал базовые возможности системы контроля версий `git` для работы с локальными репозиториями. Освоил возможность отката измененных файлов.