**МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное**

**образовательное учреждение высшего образования**

**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Отчет по лабораторной работе №2**

**Исследование основных возможностей Git и GitHub**

**по дисциплине «Основы кроссплатформенного программирования»**

|  |
| --- |
| Выполнил студент группы ИВТ-б-о-20-1 |
| Токарев В. А. « »\_\_\_\_\_\_\_\_\_20\_\_г. |
| Подпись студента\_\_\_\_\_\_\_\_\_\_\_ |
| Работа защищена « »\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г. |
| Проверил Воронкин Р. А. \_\_\_\_\_\_\_\_  (подпись) |

Ставрополь 2021

**Цель работы:** исследование базовых возможностей по работе с

локальными и удаленными ветками Git.

**Ход работы:**

Ссылка на репозиторий: <https://github.com/UnDeR-The-mAsK/lab2.git>

1. Создал новый репозиторий, сделал его удаленное клонирование

командой git clone.

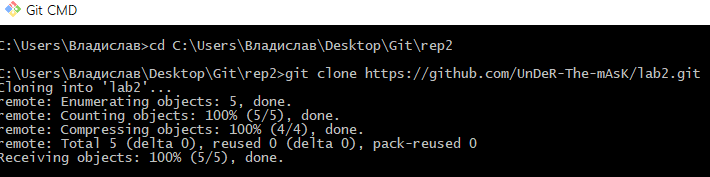


Рис. 1 Клонирование репозитория на ПК

2. Проиндексировал первый файл и сделать коммит с комментарием

"add 1.txt file". Аналогично проиндексировал и сделал коммит для других

файлов.

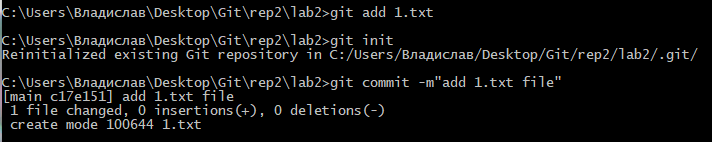


Рис. 2 Коммит для первого файла

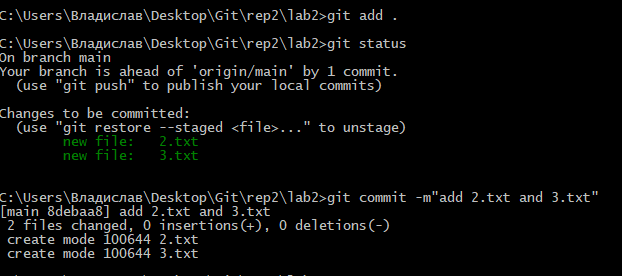


Рис. 3 Коммит для двух других файлов

3. Создал новую ветку и перешёл на неё.

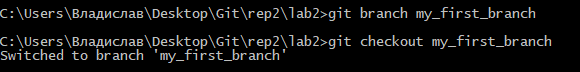


Рис. 4 Новая ветка

4. Добавил файл в новую ветку и сделал коммит.

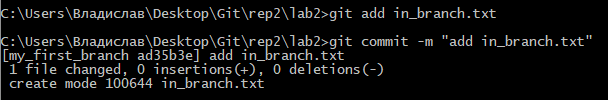


Рис. 5 Создание нового файла в ветке

5. Вернулся на ветку main и создал в ней новую, переключившись на неё.

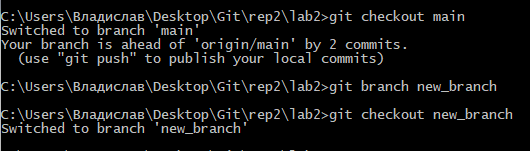


Рис. 6 Создание новой ветки

6. Сделал изменения в файле 1.txt и закоммитил его.

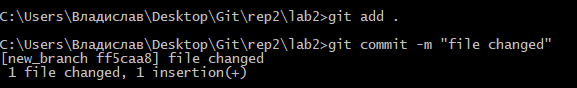


Рис. 7 Изменения в файле

7. Перешёл на ветку main и слил ветки my\_first\_branch и new\_branch.

Затем удалил их.

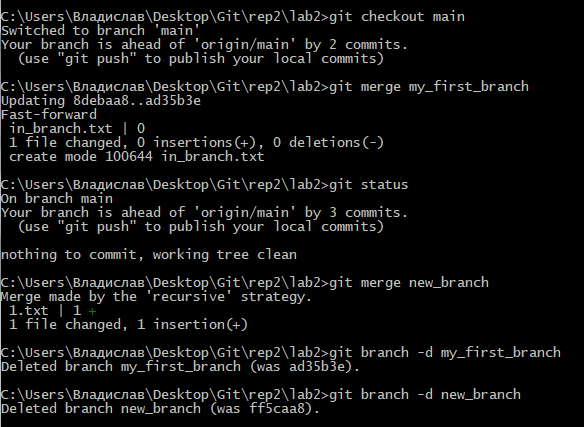


Рис. 8 Слияние веток

8. Создал ветки branch\_1 и branch\_2. Изменение в branch\_1 файлов,

также изменение в ветки branch\_2.

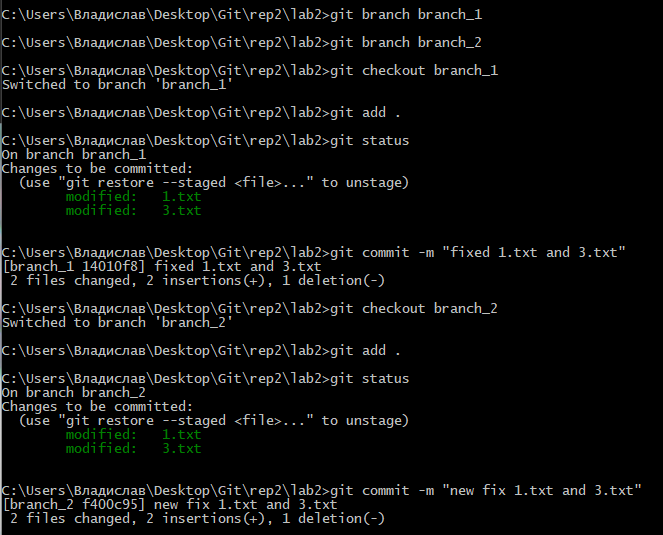


Рис. 9 Добавление новых веток и изменение в них файлов

9. Сливаем изменение ветки branch\_2 в branch\_1 и получаем конфликт.

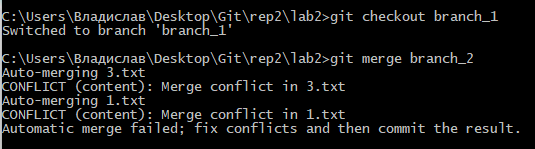


Рис. 10 Появление конфликта

10. Исправил конфликт вручную.

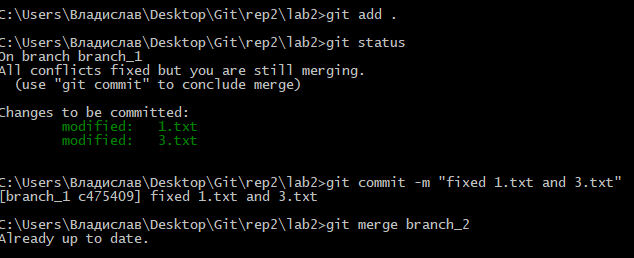


Рис. 11 Решение конфликтов

11. Отправил ветку branch\_1 на GitHub и удалил branch\_2.

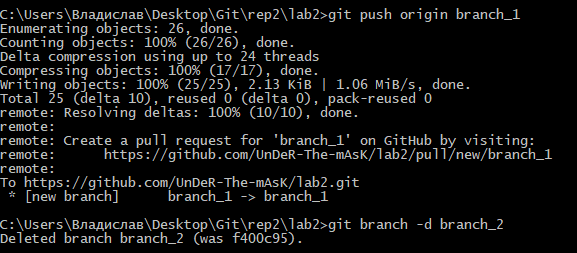


Рис. 12 Пуш ветки на GitHub

12. Средствами GitHub создал удаленную ветку branch\_3.

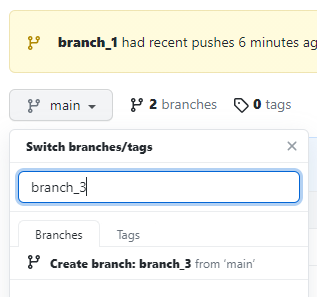


Рис. 13

13. Получил ветку branch\_3 на локальный репозиторий с помощью git fetch и создал ветку отслеживания.

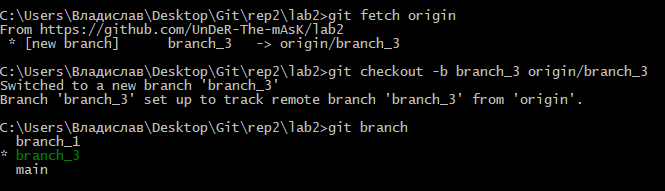


Рис. 14 Получение данных с GitHub

14. Выполнил перемещение ветки main на ветку branch\_2.

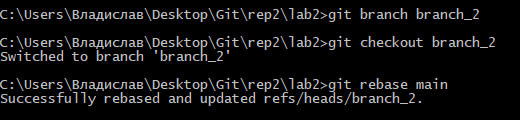


Рис. 15 Перемещение веток

15. Отправил измененные ветки на GitHub.

**Контрольные вопросы**

1. Что такое ветка?

Ветка в Git — это простой перемещаемый указатель на один из таких

коммитов. По умолчанию, имя основной ветки в Git — master

2. Что такое HEAD?

HEAD – это указатель, задача которого ссылаться на определенный

коммит в репозитории.

3. Способы создания веток.

Чтобы создать ветку необходимо ввести команду git branch и название

ветки в командную строку GidCmd

4. Как узнать текущую ветку?

Текущая ветка будет выделяться звездой в командной строке GitCmd

5. Как переключаться между ветками?

Для переключения между ветками можно ввести команду git checkout и

ввести название самой ветки.

6. Что такое удаленная ветка?

Удалённые ветки – это ссылки удалённых репозиториях пользователя,

включая ветки, теги и так далее. Полный список удалённых ссылок можно

получить с помощью команды git ls-remote или команды git remote show для

получения удалённых веток и дополнительной информации

7. Что такое ветка отслеживания?

Ветки отслеживания — это локальные ветки, которые напрямую

связаны с удалённой веткой. Если, находясь на ветке слежения, выполнить

git pull , то Git уже будет знать с какого сервера получать данные и какую

ветку использовать для слияния.

8. Как создать ветку отслеживания?

Чтобы создать ветку отслеживания нужно воспользоваться

параметрами -u или –set-upstream-to для команды git branch;

9. Как отправить изменения из локальной ветки в удаленную ветку?

Чтобы отправить изменения из локальной ветки в удалённую

необходимо ввести команду git push;

10. В чем отличие команд git fetch и git pull?

Команда git fetch забирает изменения из указанного удалённого

репозитория

А команда git pull работает как две команды (git fetch и git merge), то

есть забирает изменения из указанного удалённого репозитория, а затем

пытается слить их с предыдущей веткой git push с параметром --d (--delete).

11. Как удалить локальную и удаленную ветки?

Чтобы удалить локальную ветку достаточно просто удалить файлы,

клонированные из удалённой. Однако, чтобы удалить удалённую ветку

репозитория, нужно прописать команду git branch –d;

12. Изучить модель ветвления git-flow (использовать материалы

статей https://www.atlassi an.com/ru/git/tutorials/comparing-workflows/gitflow-

workflow, https://habr.com/ru/post/10 6912/). Какие основные типы веток

присуствуют в модели git-flow? Как организована работа с ветками в модели

git-flow? Вчемнедостаткиgit-flow?

Типы ветвей:

– master– ветка, которая стоит по умолчанию;

– develop– ветка, которая создана для ответвления от ветки masterи

слияния в неё изменений, которые пользователь собирается сливать в master

для;

Основные типы веток, которые используются в gitflow:

– Feature branches;

– Release branches;

– Hotfix branches;

Основные недостатки git flow: может замедлять работу; замедляет

просмотр реальной работы из-за кучи merge commit’a; сложно делать релизы

раньше одной недели; проблематичный в CI/CDсценариях; большие могут

затратить несколько дней на решение конфликтов.

13. На прошлой лабораторной работе было задание выбрать одно из

программных средств с GUI для работы с Git. Необходимо в рамках этого

вопроса привести описание инструментов для работы с ветками Git,

предоставляемых этим средство

Ветка в Git — это простой перемещаемый указатель на один из таких

коммитов. По умолчанию, имя основной ветки в Git — master

Sourcetree позволяет работать с теми же инструментами, которые

представление в строке CMD, но при этом управление здесь намного

интуитивно понятнее и проще. SourceTree и ветвление git-flow позволяют

убрать беспорядок в репозиториях.

⎯ Чтобы создать новую ветку, нужно нажать “branch” в верхнем

меню. В поле “new branch” ввести имя новой ветки. Выбрать пункт “create

branch”.

⎯ Если нажать на “history” в SourceTree можно увидеть, что в

репозитории теперь есть незафиксированные изменения.

⎯ Чтобы зафиксировать изменения в файле нужно нажать кнопку

“commit” вверху, чтобы зафиксировать файл. Когда откроется окно

сообщения нужно написать про свои изменения и подтвердить свои

действия, нажав на “ОК”.

⎯ Чтобы переместить данную ветку в репозитория необходимо

нажать на кнопку “push”. После этого появится диалоговое окно, в котором

нужно будет выбрать новую ветвь, чтобы указать перемещение этой ветви в

исходную точку и после этого подтвердить свои действия, нажав на “OK”

**Вывод:** в ходе выполнения лабораторной работы исследовал базовые

возможности по работе с локальными и удаленными ветками Git.