

_УП_1

Работа с Git

Подготовка

Если git никогда ранее не использовался пользователем, то необходимо установить имя и электронную почту. Для этого требуется выполнить команды, показанные на рисунке 1.

```
Korra@Korra MINGW64 ~  
$ git config --global user.name "Asad Rahimov"  
  
Korra@Korra MINGW64 ~  
$ git config --global user.email "abramovaleva3007@gmail.com"  
  
Korra@Korra MINGW64 ~  
$ |
```

Рисунок 1 - Установка имени и электронной почты

Затем нужно указать параметры установки окончаний строк (рисунок 2).

```
Korra@Korra MINGW64 ~  
$ git config --global core.autocrlf true  
  
Korra@Korra MINGW64 ~  
$ git config --global core.safecrlf true  
  
Korra@Korra MINGW64 ~  
$ |
```

Рисунок 2 - Параметры окончаний строк

И последним пунктом идет установка отображения Unicode, показанная на рисунке 3.

```
Korra@Korra MINGW64 ~  
$ git config --global core.quotePath off  
  
Korra@Korra MINGW64 ~  
$
```

Рисунок 3 - Установка отображения unicode

Создание проекта

Сначала нужно создать пустой каталог и внутри него файл `hello.html` (рисунок 4).

```
Korra@Korra MINGW64 ~/my_project (master)
$ mkdir hello

Korra@Korra MINGW64 ~/my_project (master)
$ cd hello

Korra@Korra MINGW64 ~/my_project/hello (master)
$ touch hello.html

Korra@Korra MINGW64 ~/my_project/hello (master)
$
```

Рисунок 4 - Создание каталога и файла

После этого в файл необходимо ввести данные, например, «Hello, world», как показано на рисунке 5.

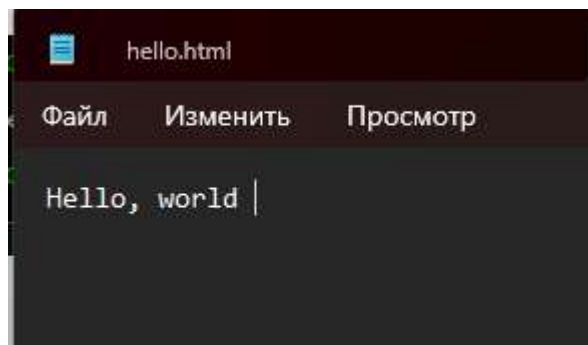


Рисунок 5 - Содержание файла

Для создания репозитория

```
Korra@Korra MINGW64 ~/my_project
$ git init
Initialized empty Git repository in C:/Users/Korra/my_project/.git/
```

используется команда `git init` (рисунок 6).

Рисунок 6 - Создание репозитория

Для добавления страницы в репозиторий необходима команда `git add` (рисунок 7).

Рисунок 7 - Добавление в репозиторий

```
Korra@Korra MINGW64 ~/my_project/hello (master)
$ git commit -m "First commit"
[master (root-commit) 2be9ca2] First commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 hello/hello.html
```

Проверка состояния

Проверка состояния репозитория осуществляется с помощью команды `git status`. Если в репозитории хранится текущее состояние рабочего каталога и нет изменений, ожидающих записи, будет показано сообщение, как на рисунке 8.

```
Korra@Korra MINGW64 ~/my_project/hello (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Рисунок 8 - Проверка состояния репозитория

Внесение изменений

Сначала необходимо внести изменения в файл (рисунок 9).

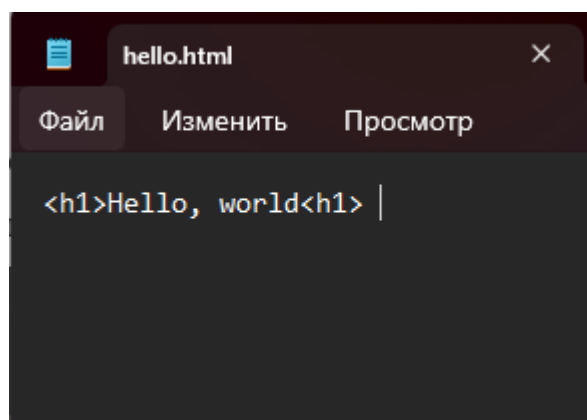


Рисунок 9 - Внесение изменений в файл

Если после предыдущего пункта осуществить проверку состояния репозитория, то будет показано данное сообщение (рисунок 10).

Рисунок 10 - Сообщение о незафиксированных изменениях

```
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        a.html
        b.html
        hello.html
        v.html

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

Индексация изменений

Для того, чтобы проиндексировать изменения, нужно осуществить действия, показанные на рисунке 11.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello.html
```

Рисунок 11 - Команды для индексации изменений

После этого изменения файлы были проиндексированы. Это значит, что пока изменения не записаны в репозиторий. Если изменения позже не нужно будет фиксировать, то индексацию можно снять командой `git reset`.

Индексация и коммит

Можно зафиксировать изменения отдельными коммитами. Как это сделать, показано на рисунках 12-14.

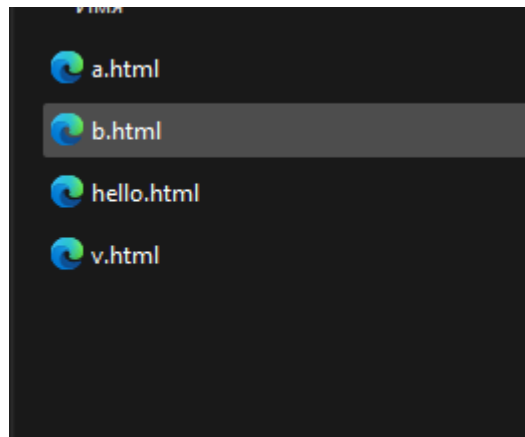


Рисунок 12 - Создано 3 файла

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Changes for a and b"
[master (root-commit) 75a79c0] Changes for a and b
3 files changed, 1 insertion(+)
create mode 100644 a.html
create mode 100644 b.html
```

Рисунок 13 - Индексация и коммит для 2 файлов

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Unrelated change for v"
[master 629d2a4] Unrelated change for v
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 v.html
```

Рисунок 14 - Индексация и коммит для третьего файла

Коммит изменений

Для того, чтобы редактировать комментарий коммита, нужно использовать команду `git commit` без метки `-m`.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit
On branch master
nothing to commit, working tree clean
```

Рисунок 15 - Коммит изменений

После этого будет открыт редактор, в котором необходимо на 1 строке записать комментарий коммита (рисунок 16).

Рисунок 16 - Ввод комментария

После выхода из текстового редактора будет указано следующее сообщение (рисунок 17).

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git commit
hint: Waiting for your editor to close the file...
[main 2021-01-31T08:05:41.046Z] update#setState idle
[main 2021-01-31T08:06:11.084Z] update#setState checking for updates
[main 2021-01-31T08:06:11.304Z] update#setState idle
[master c8f53f9] Added h2 tag
1 file changed, 2 insertions(+), 1 deletion(-)

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ |
```

Рисунок 17 - Коммит-сообщения

После этого еще раз нужно проверить состояние репозитория (рисунок 18).

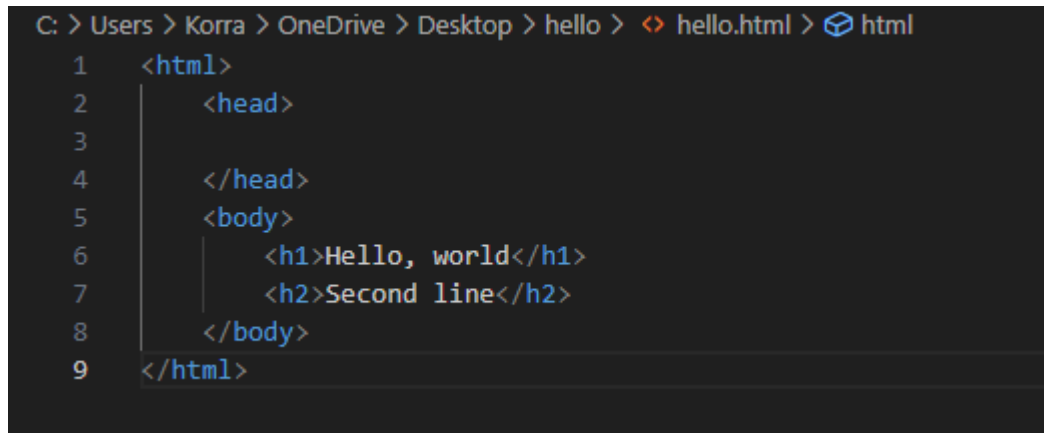
```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html
```

Рисунок 18 - Проверка состояния

Изменения, а не файлы

Для того, чтобы понять, что git фокусируется на изменениях в файле, а не на самом файле, можно проделать следующие действия.

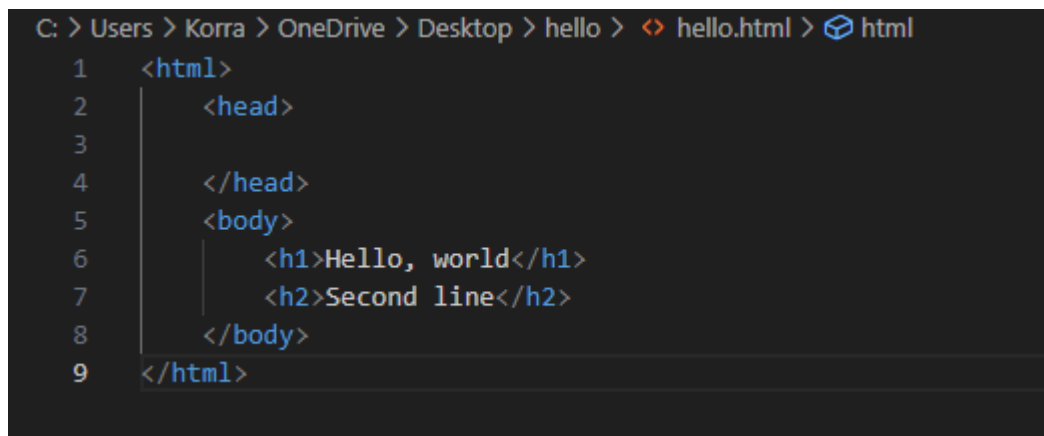
Сначала нужно в файл hello.html добавить теги html и body (рисунок 19), а затем проиндексировать изменения.



```
C: > Users > Korra > OneDrive > Desktop > hello > <img alt="git icon" data-bbox="445 215 460 230"/> hello.html > <img alt="html icon" data-bbox="715 215 730 230"/> html
1  <html>
2    <head>
3
4    </head>
5    <body>
6      <h1>Hello, world</h1>
7      <h2>Second line</h2>
8    </body>
9  </html>
```

Рисунок 19 - Добавление тегов html и body

Затем еще раз нужно добавить изменения в файл (добавить тег head), но изменения не индексировать (рисунок 20).



```
C: > Users > Korra > OneDrive > Desktop > hello > <img alt="git icon" data-bbox="625 498 640 513"/> hello.html > <img alt="html icon" data-bbox="715 498 730 513"/> html
1  <html>
2    <head>
3
4    </head>
5    <body>
6      <h1>Hello, world</h1>
7      <h2>Second line</h2>
8    </body>
9  </html>
```

Рисунок 20 - Добавление тега head

Далее нужно проверить статус. На рисунке 21 видно, что файл hello.html указан дважды: первое изменение проиндексировано и готово к коммиту, а второе — нет.

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

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:   hello.html

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$
```

Рисунок 21 - Проверка состояния

Далее надо произвести коммит проиндексированного изменения и затем еще раз проверить состояние (рисунок 22).

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git commit -m "Add standard HTML tags"
[master cab488a] Add standard HTML tags
1 file changed, 6 insertions(+), 2 deletions(-)

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ 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:   hello.html

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

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ |
```

Рисунок 22 - Коммит 1 изменения и проверка состояния

Нужно добавить второе изменение в индекс и затем проверить состояние (рисунок 23).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git add .

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html
```

Рисунок 23 - Индексация 2 изменения и проверка состояния

После этого нужно сделать коммит второго изменения (рисунок 24).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Add HTML header"
[master f12f0a3] Add HTML header
1 file changed, 9 insertions(+), 1 deletion(-)
```

Рисунок 24 - Коммит 2 изменения

_УП_2

Работа с Git

История

Для того, чтобы просмотреть список произведенных изменений в проекте, используется команда `git log` (рисунок 1).

```

git log
commit f12f0a3fd45f1871d28523cf04580586c60394bd (HEAD -> master)
Author: Asad Rahimov <abramovaleva3007@gmail.com>
Date: Mon Jun 2 17:14:22 2025 +0300

    Add HTML header

commit 629d2a4140e6883379cfadae552c01b400a9bcb6
Author: Asad Rahimov <abramovaleva3007@gmail.com>
Date: Mon Jun 2 17:06:09 2025 +0300

    Unrelated change for v

commit 75a79c092bf1d50716f21277543dad8e091b77d0
Author: Asad Rahimov <abramovaleva3007@gmail.com>
Date: Mon Jun 2 17:04:51 2025 +0300

    Changes for a and b

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |

```

Рисунок 25 - Просмотр истории изменений

На рисунке 1 была выведена полная история. Для того, чтобы увидеть однострочный формат используется команда `git log --pretty=oneline` (рисунок 2).

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --pretty=oneline --max-count=2
f12f0a3fd45f1871d28523cf04580586c60394bd (HEAD -> master) Add HTML header
629d2a4140e6883379cfadae552c01b400a9bcb6 Unrelated change for v

```

Рисунок 26 - Однострочный формат вывода

Далее на рисунках 3-8 показано несколько вариантов вывода истории изменений.

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --pretty=oneline --until='5 minutes ago'
f12f0a3fd45f1871d28523cf04580586c60394bd (HEAD -> master) Add HTML header
629d2a4140e6883379cfadae552c01b400a9bcb6 Unrelated change for v
75a79c092bf1d50716f21277543dad8e091b77d0 Changes for a and b

```

Рисунок 27 - Вывод последних 2 изменений

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --pretty=oneline --since '10 days ago'
0eb51bd098cba74b2bfde4f74631a1dd261b78b2 (HEAD -> master, style) Force new commit for rebase
b96818bcc2ec55f59c3d4bda57a016bb1526844b Life is Horrible
6f500841f1735ac6253ccf25a349af3c03b06f93 Added README
4d3884af8fbaa889b08f65a707c2aa26d12d6f28 Update index.html
869858db14be53d4e4616bd54662f3e85f3d123d Added css stylesheet
1b5a3ec0d1f593810770da10524ee0a1735fd7e9 Added index.html
42249ded2bf42c6c801556770e66667c4c04cc20 Moved hello.html to lib
cad1cec19b44043f5d9adb68da062dd1f94c2a5a Add an author/email comment
629d2a4140e6883379cfadae552c01b400a9bcb6 Unrelated change for v
75a79c092bf1d50716f21277543dad8e091b77d0 Changes for a and b
```

Рисунок 28 - Вывод изменений начиная с определенного времени

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --pretty=oneline --until='5 minutes ago'
f12f0a3fd45f1871d28523cf04580586c60394bd (HEAD -> master) Add HTML header
629d2a4140e6883379cfadae552c01b400a9bcb6 Unrelated change for v
75a79c092bf1d50716f21277543dad8e091b77d0 Changes for a and b
```

Рисунок 29 - Вывод изменений до определенного времени

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --pretty=oneline --author='Asad Rahimov'
f12f0a3fd45f1871d28523cf04580586c60394bd (HEAD -> master) Add HTML header
629d2a4140e6883379cfadae552c01b400a9bcb6 Unrelated change for v
75a79c092bf1d50716f21277543dad8e091b77d0 Changes for a and b
```

Рисунок 30 - Вывод изменений, внесенных определенным автором

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --pretty=oneline --all
f12f0a3fd45f1871d28523cf04580586c60394bd (HEAD -> master) Add HTML header
629d2a4140e6883379cfadae552c01b400a9bcb6 Unrelated change for v
75a79c092bf1d50716f21277543dad8e091b77d0 Changes for a and b
```

Рисунок 31 - Вывод всех изменений

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --all --pretty=format:"%h %cd %s (%an)" --since '12 days ago'
0eb51bd Mon Jun 2 19:56:48 2025 +0300 Force new commit for rebase (Asad Rahimov)
b96818b Mon Jun 2 19:33:34 2025 +0300 Life is Horrible (Asad Rahimov)
6f50084 Mon Jun 2 19:19:36 2025 +0300 Added README (Asad Rahimov)
4d3884a Mon Jun 2 19:05:38 2025 +0300 Update index.html (Asad Rahimov)
869858d Mon Jun 2 18:48:56 2025 +0300 Added css stylesheet (Asad Rahimov)
1b5a3ec Mon Jun 2 18:31:22 2025 +0300 Added index.html (Asad Rahimov)
42249de Mon Jun 2 18:26:29 2025 +0300 Moved hello.html to lib (Asad Rahimov)
cad1cec Mon Jun 2 18:15:42 2025 +0300 Add an author/email comment (Asad Rahimov)
f12f0a3 Mon Jun 2 17:14:22 2025 +0300 Add HTML header (Asad Rahimov)
629d2a4 Mon Jun 2 17:06:09 2025 +0300 Unrelated change for v (Asad Rahimov)
75a79c0 Mon Jun 2 17:04:51 2025 +0300 Changes for a and b (Asad Rahimov)
```

Рисунок 32 - Использование нескольких параметров

Алиасы

Для настройки алиасов используется команда, показанная на рисунке 9.

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.co checkout

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.ci commit

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.st status

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.br branch

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.hist "log --pretty=format:'%h %ad | %s%d [%an]' --graph --date=short"

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.type 'cat-file -t'

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.dump 'cat-file -p'

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$

```

Рисунок 33 - Настройка алиасов для некоторых команд

При выполнении алиаса будет выполнена определенная команда и выведены нужные данные (рисунок 10).

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git config --global alias.hist "log --pretty=format:'%h %ad | %s%d [%an]' --graph --date=short --all"

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git hist
* 0eb51bd 2025-06-02 | Force new commit for rebase (HEAD -> master, style) [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

```

Рисунок 34 - Выполнение алиаса hist

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ alias gs='git status '

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ gs
On branch master
nothing to commit, working tree clean

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
```

Рисунок 35 - Установка и выполнение алиаса gs

Получение старых версий

Для того, чтобы вернуть рабочий каталог к предыдущему состоянию, можно использовать следующий способ: для начала нужно узнать хэши предыдущих версий, что можно сделать с помощью ранее заданного алиаса hist (рисунок 12).

```
$ git hist
* 0eb51bd 2025-06-02 | Force new commit for rebase (HEAD -> master, style) [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
```

Рисунок 36 - Просмотр хэшей предыдущих версий

Далее нужно выполнить команду git checkout с номером нужного хэша (достаточно первых 7 знаков). После этого можно просмотреть содержимое файла с помощью команды cat (рисунок 13).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ cat hello.html
hello, world
```

Рисунок 37 - Возвращение к нужной версии и просмотр содержимого файла

Возвращение к последней версии в ветке master

Для возвращения к последней версии в ветке master (имя ветки по умолчанию) надо ввести команду git checkout master, что показано на рисунке 14.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ cat hello.html
<html>
  <head>

  </head>
  <body>
    <h1>Hello, world</h1>
    <h2>Second line</h2>
  </body>
</html>
```

Рисунок 38 - Возвращение к последней версии в ветке master

Создание тегов версий

Для создания тега используется команда `git tag`. На рисунке 15 показано, тегом `ver1` была названа текущая версия страницы.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git tag ver1
```

Рисунок 39 - Задание тега

Чтобы перейти к предыдущей версии, можно использовать символ «`^`», который означает «родитель».

```
You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 629d2a4 Unrelated change for v

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((629d2a4...))
$ cat hello.html
<h1>Hello,world<h1> yutjyfhj
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((629d2a4...))
$
```

Рисунок 40 – Переход к предыдущей версии с помощью тега

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git tag ver1-beta
```

Рисунок 41 - Задание тега предыдущей версии

Теперь с помощью тегов можно переключаться между версиями (рисунок 18).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((629d2a4...))
$ git checkout ver1
Previous HEAD position was 629d2a4 Unrelated change for v
HEAD is now at f12f0a3 Add HTML header

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((ver1))
$ git checkout ver1-beta
HEAD is now at f12f0a3 Add HTML header

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((ver1))
$
```

Рисунок 42 - Переключение между версиями с помощью тегов

Для просмотра всех тегов используется команда `git tag` (рисунок 19).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((ver1))
$ git tag
ver1
ver1-beta
```

Рисунок 43 - Просмотр тегов

Также можно просмотреть теги в логе, как показано на рисунке 20.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((ver1))
$ git hist master --all
* 0eb51bd 2025-06-02 | Force new commit for rebase (style, master) [Asad Rahimov]
]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (HEAD, tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
```

Рисунок 44 - Просмотр тегов в логе

Отмена локальных изменений (до индексации)

Сначала нужно переключиться на последний коммит `master` (рисунок 21).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello ((ver1))
$ git checkout master
M      hello.html
Switched to branch 'master'
```

Рисунок 45 - Переключение на последний коммит

Далее для работы нужно внести изменение в файл (рисунок 22).


```

C: > Users > Korra > OneDrive > Desktop > hello > <> hello.html > html > ht
1  <html>
2      <head>
3
4      </head>
5      <body>
6          <h1>Hello, world</h1>
7          <h2>Second line</h2>
8          <!-- Это плохой комментарий (очень плохой!) -->
9      </body>
10 </html>

```

Рисунок 46 - Внесение изменения в файл

После выполнения команды `git status` будет показано, что есть не проиндексированное изменение (рисунок 23).

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git st
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:   hello.html

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

```

Рисунок 47 - Не проиндексированное изменение

Для переключения в версию файла без изменений используется команда `git checkout hello.html` (рисунок 24). Команда `git status` покажет, что не было произведено изменений, не зафиксированных в каталоге.

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git checkout hello.html
Updated 1 path from the index

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git st
On branch master
nothing to commit, working tree clean

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ cat hello.html
<html>
  <head>

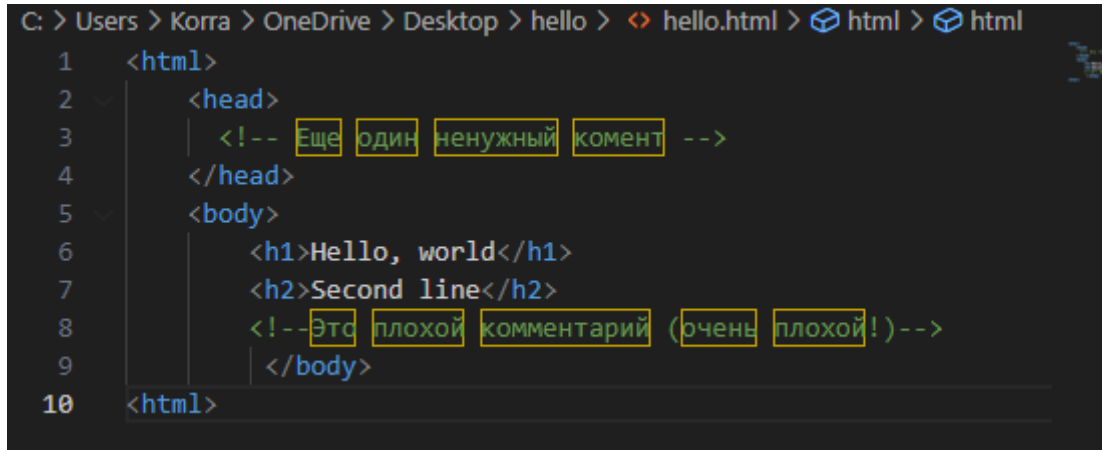
  </head>
  <body>
    <h1>Hello, world</h1>
    <h2>Second line</h2>
  </body>
</html>
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)

```

Рисунок 48 - Возвращение к версии

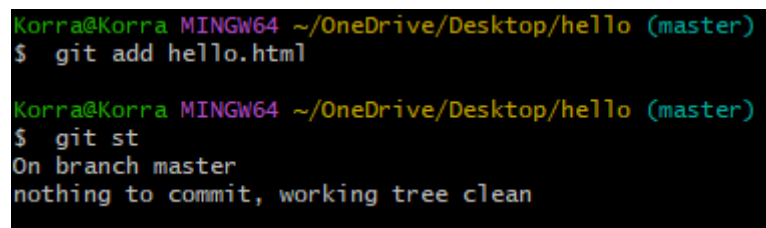
Отмена проиндексированных изменений (перед коммитом)

Для того, чтобы научиться отменять проиндексированные изменения, сначала нужно внести ненужное изменение в файл (рисунок 25). После этого производится индексация (рисунок 26).



```
C: > Users > Korra > OneDrive > Desktop > hello > hello.html > html > html
1 <html>
2   <head>
3     <!-- Еще один ненужный комент -->
4   </head>
5   <body>
6     <h1>Hello, world</h1>
7     <h2>Second line</h2>
8     <!-- Это плохой комментарий (очень плохой!) -->
9   </body>
10 </html>
```

Рисунок 49 - Внесение ненужного изменения

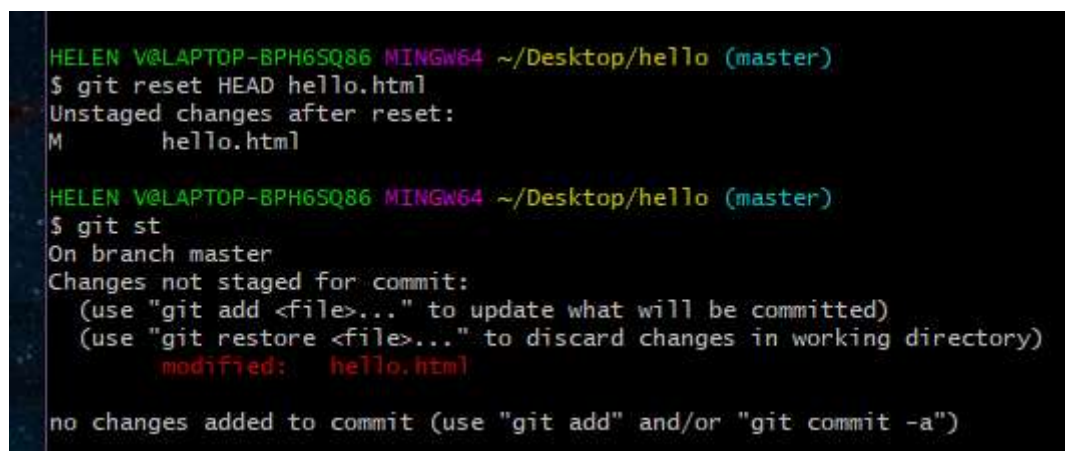


```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git add hello.html

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git st
On branch master
nothing to commit, working tree clean
```

Рисунок 50 - Индексация изменения

Для отмены индексация изменения используется команда `git reset HEAD hello.html` (рисунок 27). Команда `reset` сбрасывает буферную зону к HEAD и очищает ее от проиндексированных изменений. Но для удаления ненужного по-прежнему используется команда `git checkout` (рисунок 28).



```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git reset HEAD hello.html
Unstaged changes after reset:
M   hello.html

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git st
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:   hello.html

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

Рисунок 51 - Очистка буферной зоны

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git checkout hello.html
Updated 0 paths from the index

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git st
On branch master
nothing to commit, working tree clean

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |

```

Рисунок 52 - Удаление ненужных изменений

Отмена коммитов

Для отмены коммита можно использовать способ создания нового коммита, отменяющего изменения.

Для начала надо внести изменение, проиндексировать его и записать коммит (рисунки 29-30).

```

4      </head>
5      <body>
6          <h1>Hello, world</h1>
7          <h2>Second line</h2>
8          <!--опять комментарий.....-->
9          </body>
10     <html> |

```

Рисунок 53 - Внесение изменения в файл

```

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git add hello.html

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git commit -m "Oops, this is bad commit"
[master ce12d52] Oops, this is bad commit
1 file changed, 1 insertion(+)

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ |

```

Рисунок 54 - Индексация и коммит

Для создания коммита, который удалит ненужные изменения, используется команда `git revert HEAD` (рисунок 31). После этого будет открыт редактор, в котором можно отредактировать коммит сообщение (рисунок 32), затем надо сохранить файл и закрыть редактор (рисунок 33).

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git revert HEAD
hint: Waiting for your editor to close the file...
(electron) Sending uncompressed crash reports is deprecated and will be removed in a future version of Electron. Set { compress: true } to opt-in to the new behavior. Crash reports will be uploaded gzipped, which most crash reporting servers support.
[main 2021-02-13T20:03:20.025Z] update#setState idle
(node:11232) electron: The default of contextIsolation is deprecated and will be changing from false to true in a future release of Electron. See https://github.com/electron/electron/issues/23506 for more information
(node:11232) electron: The default of contextIsolation is deprecated and will be changing from false to true in a future release of Electron. See https://github.com/electron/electron/issues/23506 for more information
```

Рисунок 55 - Выполнение команды git revert

```
Revert "Oops, this is bad commit"

This reverts commit ce12d52b0101f30fefa7c66203593dbfac57ae54.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# Changes to be committed:
#   modified:   hello.html
#
```

Рисунок 56 - Коммит сообщение в редакторе

```
(node:11232) electron: The default of contextIsolation is deprecated and will be changing from false to true in a future release of Electron. See https://github.com/electron/electron/issues/23506 for more information
[main 2021-02-13T20:03:50.027Z] update#setState checking for updates
[main 2021-02-13T20:03:50.085Z] update#setState idle
[master e04a55f] Revert "Oops, this is bad commit"
1 file changed, 1 deletion(-)

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ |
```

Рисунок 57 - Редактор закрыт

При проверке лога будут показаны все коммиты, в том числе и отмененные (рисунок 34).

```
1 file changed, 1 deletion(-)
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git hist
* e04a55f 2021-02-13 | Revert "Oops, this is bad commit" (HEAD -> master) [Elena Verstova]
* ce12d52 2021-02-13 | Oops, this is bad commit [Elena Verstova]
* d653372 2021-01-31 | Add HTML header (tag: ver1) [Elena Verstova]
* cab488a 2021-01-31 | Add standard HTML tags (tag: ver1-beta) [Elena Verstova]
* c8f53f9 2021-01-31 | Added h2 tag [Elena Verstova]
* cf86455 2021-01-31 | Unrelated change for v [Elena Verstova]
* 00106a5 2021-01-31 | Changes for a and b [Elena Verstova]
* 9234a79 2021-01-31 | First Commit [Elena Verstova]

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$
```

Рисунок 58 - Все коммиты при просмотре лога

Перед удалением коммита последний из них нужно отметить тегом, чтобы не потерять его (рисунок 35).

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git tag oooops

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git hist
* e04a55f 2021-02-13 | Revert "Oops, this is bad commit" (HEAD -> master, tag: oooops) [Elena Verstova]
* ce12d52 2021-02-13 | Oops, this is bad commit [Elena Verstova]
* d653372 2021-01-31 | Add HTML header (tag: ver1) [Elena Verstova]
* cab488a 2021-01-31 | Add standard HTML tags (tag: ver1-beta) [Elena Verstova]
* c8f53f9 2021-01-31 | Added h2 tag [Elena Verstova]
* cf86455 2021-01-31 | Unrelated change for v [Elena Verstova]
* 00106a5 2021-01-31 | Changes for a and b [Elena Verstova]
* 9234a79 2021-01-31 | First Commit [Elena Verstova]

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$
```

Рисунок 59 - Отметка тегом коммита

Для сброса коммитов используется команда `git reset --hard ver1` (рисунок 36). Она сбрасывает ветку до версии с тегом `ver1`.

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git reset --hard ver1
HEAD is now at d653372 Add HTML header

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git hist
* d653372 2021-01-31 | Add HTML header (HEAD -> master, tag: ver1) [Elena Verstova]
* cab488a 2021-01-31 | Add standard HTML tags (tag: ver1-beta) [Elena Verstova]
* c8f53f9 2021-01-31 | Added h2 tag [Elena Verstova]
* cf86455 2021-01-31 | Unrelated change for v [Elena Verstova]
* 00106a5 2021-01-31 | Changes for a and b [Elena Verstova]
* 9234a79 2021-01-31 | First Commit [Elena Verstova]

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$
```

Рисунок 60 - Сброс коммита

Но при просмотре лога с помощью команды `git hist --all` отмененные коммиты по-прежнему будут показываться, так как они всё еще находятся в репозитории (рисунок 37).

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git hist --all
* e04a55f 2021-02-13 | Revert "Oops, this is bad commit" (tag: oooops) [Elena Verstova]
* ce12d52 2021-02-13 | Oops, this is bad commit [Elena Verstova]
* d653372 2021-01-31 | Add HTML header (HEAD -> master, tag: ver1) [Elena Verstova]
* cab488a 2021-01-31 | Add standard HTML tags (tag: ver1-beta) [Elena Verstova]
* c8f53f9 2021-01-31 | Added h2 tag [Elena Verstova]
* cf86455 2021-01-31 | Unrelated change for v [Elena Verstova]
* 00106a5 2021-01-31 | Changes for a and b [Elena Verstova]
* 9234a79 2021-01-31 | First Commit [Elena Verstova]

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$
```

Рисунок 61 - Сброшенные коммиты находятся по-прежнему в репозитории

Удаление тега

Так как тег «oooops» больше не нужен, его и коммиты, на которые он указывает, можно удалить с помощью команды `git tag -d` (рисунок 38).

```
HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git tag -d oooops
Deleted tag 'oooops' (was e04a55f)

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$ git hist --all
* d653372 2021-01-31 | Add HTML header (HEAD -> master, tag: ver1) [Elena Verstova]
* cab488a 2021-01-31 | Add standard HTML tags (tag: ver1-beta) [Elena Verstova]
* c8f53f9 2021-01-31 | Added h2 tag [Elena Verstova]
* cf86455 2021-01-31 | Unrelated change for v [Elena Verstova]
* 00106a5 2021-01-31 | Changes for a and b [Elena Verstova]
* 9234a79 2021-01-31 | First Commit [Elena Verstova]

HELEN V@LAPTOP-BPH6SQ86 MINGW64 ~/Desktop/hello (master)
$
```

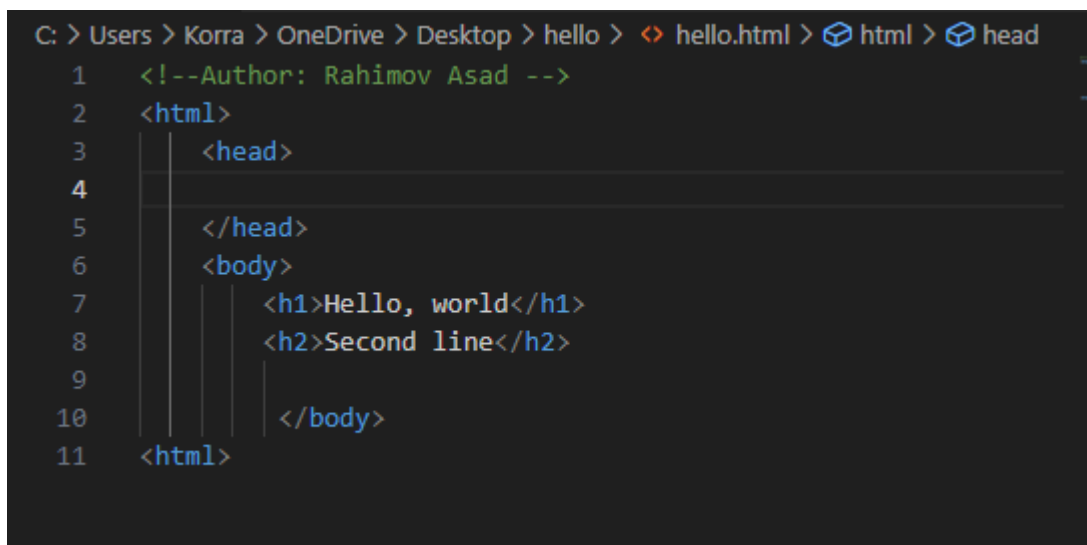
Рисунок 62 - Удаление тега

_УП_3

Работа с Git

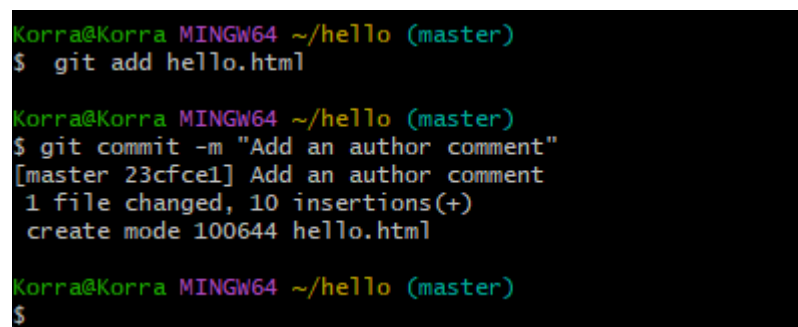
Внесение изменений в коммиты

Для начала будет создан коммит, в который позже будут внесены изменения. На рисунках 1 и 2 происходит добавление комментария в файл hello.html и его индексация и коммит.



```
C: > Users > Korra > OneDrive > Desktop > hello > <> hello.html > html > head
1  <!--Author: Rahimov Asad -->
2  <html>
3      <head>
4
5      </head>
6      <body>
7          <h1>Hello, world</h1>
8          <h2>Second line</h2>
9
10         </body>
11 </html>
```

Рисунок 63 - Добавление комментария в файл



```
Korra@Korra MINGW64 ~/hello (master)
$ git add hello.html

Korra@Korra MINGW64 ~/hello (master)
$ git commit -m "Add an author comment"
[master 23cfce1] Add an author comment
1 file changed, 10 insertions(+)
create mode 100644 hello.html

Korra@Korra MINGW64 ~/hello (master)
$
```

Рисунок 64 - Индексация и коммит

Далее необходимо добавить электронную почту в комментарий (рисунок 3).

```

C: > Users > Korra > OneDrive > Desktop > hello > <> hello.html > html > head
1  <!--Author: Rahimov Asad (abramovaleva3007@gmail.com)-->
2  <html>
3      <head>
4
5      </head>
6      <body>
7          <h1>Hello, world</h1>
8          <h2>Second line</h2>
9
10         </body>
11 </html>

```

Рисунок 65 - Добавление электронной почты

Но для того, чтобы не создавать отдельный коммит ради электронной почты, можно изменить предыдущий так, как показано на рисунке 4.

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit --amend -m "Add an author/email comment"
[master cad1cec] Add an author/email comment
Date: Mon Jun 2 17:14:22 2025 +0300
1 file changed, 9 insertions(+), 1 deletion(-)

```

Рисунок 66 - Индексация и изменение коммита

При просмотре истории можно будет заметить, что последний коммит был изменен.

```

Korra@Korra MINGW64 ~/hello (master)
$ git hist
* 23cfce1 2025-06-03 | Add an author comment (HEAD -> master) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase (style) [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/hello (master)
$ |

```

Рисунок 67 - Последний коммит изменен

Перемещение файлов

Для перемещения файлов в пределах репозитория используются команды, показанные на рисунке 6. После выполнения данных команды git

индексирует эти изменения (удаление файла hello.html и создание файла lib/hello.html).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git mv hello.html lib

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    hello.html -> lib/hello.html

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$
```

Рисунок 68 - Перемещение файла

Далее надо осуществить коммит данного перемещения.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Moved hello.html to lib"
[master 42249de] Moved hello.html to lib
1 file changed, 0 insertions(+), 0 deletions(-)
rename hello.html => lib/hello.html (100%)

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$
```

Рисунок 69 – Коммит перемещения

[Подробнее о структуре](#)

Необходимо добавить еще один файл в репозиторий. Это будет файл index.html с кодом, показанным на рисунке 8.

```
C: > Users > Korra > OneDrive > Desktop > hello > <> index.html
1  <html>
2  |   <body>
3  |   |   <iframe src="lib/hello.html" width="200" height="200"></iframe>
4  |   </body>
5  </html>
```

Рисунок 70 - Содержимое файла index.html

Далее нужно проиндексировать и закоммитить файл (рисунок 9).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Added index.html"
[master 1b5a3ec] Added index.html
1 file changed, 5 insertions(+)
create mode 100644 index.html

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 71 - Индексация и коммит

При открытии файла index.html будет виден кусок страницы hello.html (рисунок 10).

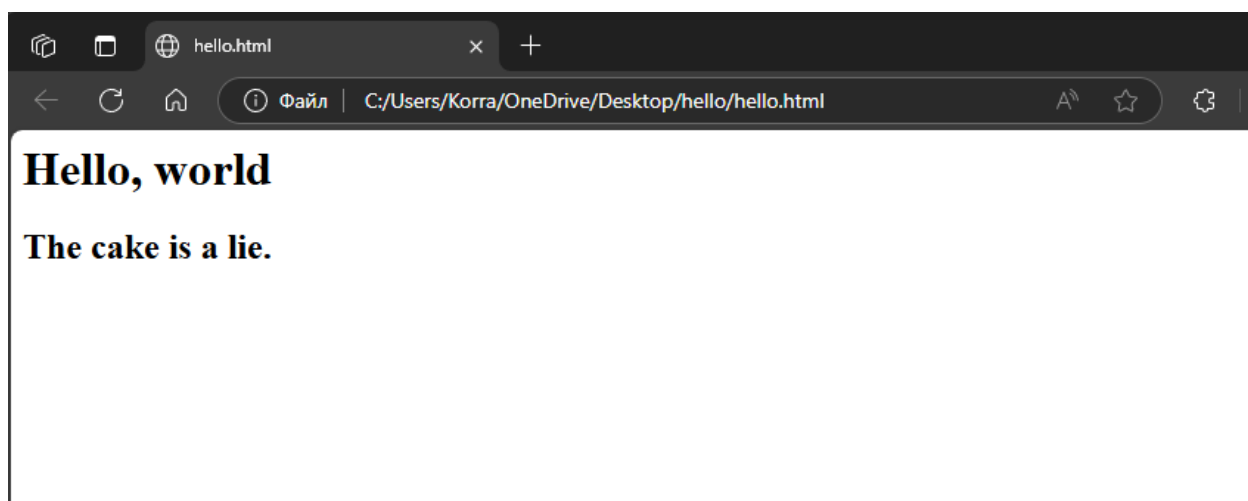


Рисунок 72 - Файл index.html, открытый в браузере

Каталог .git

Чтобы посмотреть структуру каталога .git необходимо выполнить команду, показанную на рисунке 11.

```
Korra@Korra MINGW64 ~/hello (master)
$ ls -C .git
COMMIT_EDITMSG  ORIG_HEAD  description  index  logs/  refs/
HEAD           config     hooks/       info/  objects/

Korra@Korra MINGW64 ~/hello (master)
$
```

Рисунок 73 - Содержание каталога .git

При аналогичном просмотре каталога objects можно будет увидеть множество каталогов с именами из 2 символов (рисунок 12). Имена каталогов являются первыми двумя буквами хэша.

```
Korra@Korra MINGW64 ~/hello (master)
$ ls -C .git/objects
00/ 0e/ 23/ 47/ 56/ 67/ 6f/ 80/ b2/ ca/ e6/ pack/
03/ 0f/ 26/ 4a/ 62/ 69/ 75/ 86/ b9/ cf/ f1/
04/ 1b/ 42/ 4d/ 66/ 6d/ 79/ 96/ c6/ d6/ info/

Korra@Korra MINGW64 ~/hello (master)
$
```

Рисунок 74 - Содержание каталога objects

При просмотре содержимого любого из каталогов будут показаны файлы, названия которых состоят из 38 символов (рисунок 13).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ ls -C .git/objects
03/ 16/ 34/ 49/ 57/ 6e/ 88/ ae/ bc/ d2/ e1/ fd/
07/ 1e/ 37/ 4e/ 5c/ 74/ 8d/ af/ c2/ d8/ f3/ info/
12/ 24/ 44/ 50/ 67/ 7f/ a0/ b5/ ce/ db/ f9/ pack/

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ ls -C .git/objects/8d
e3e8034c7ba4cc85a9ae26844a5c51be5f5e66

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$
```

Рисунок 75 - Просмотр каталога 8d

Далее требуется просмотреть файл конфигурации с помощью команды cat (рисунок 14).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ cat .git/config
[core]
    repositoryformatversion = 0
    filemode = false
    bare = false
    logallrefupdates = true
    symlinks = false
    ignorecase = true

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$
```

Рисунок 76 - Просмотр файла конфигурации

На рисунке 14 показан просмотр файлов в подкаталоге tags и веток в каталоге heads.

```
Korra@Korra MINGW64 ~/hello (master)
$ ls .git/refs
heads/ tags/

Korra@Korra MINGW64 ~/hello (master)
$ ls .git/refs/heads
master style

Korra@Korra MINGW64 ~/hello (master)
$ ls .git/refs/tags
ver1 ver1-beta

Korra@Korra MINGW64 ~/hello (master)
$ cat .git/refs/tags/ver1
f12f0a3fd45f1871d28523cf04580586c60394bd

Korra@Korra MINGW64 ~/hello (master)
$ |
```

Рисунок 77 - Просмотр файлов и веток

Файл HEAD содержит ссылку на текущую ветку (рисунок 16).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ cat .git/HEAD
ref: refs/heads/master

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 78 - Содержимое файла HEAD

Работа с объектами git

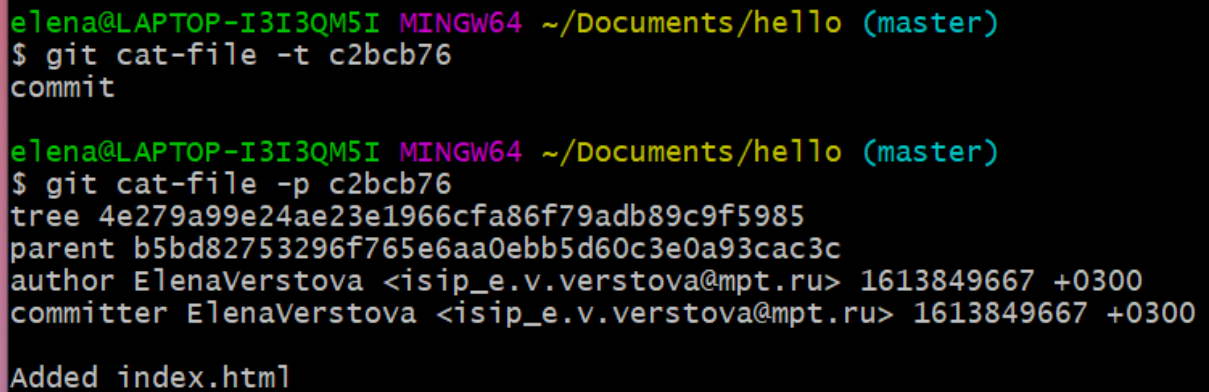
Для начала необходимо просмотреть последний коммит (рисунок 17).

```
Korra@Korra MINGW64 ~/hello (master)
$ git hist --max-count=1
* 23cfce1 2025-06-03 | Add an author comment (HEAD -> master) [Asad Rahimov]

Korra@Korra MINGW64 ~/hello (master)
```

Рисунок 79 - Последний коммит

Далее надо использовать хэш последнего коммита, используя команды `cat-file -p` и `cat-file -t` (рисунок 18) для просмотра объекта коммита. Также вместо длинных команд можно использовать сокращенные `type` и `dump`, если данные команды были заданы как алиасы.



```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git cat-file -t c2bcb76
commit

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git cat-file -p c2bcb76
tree 4e279a99e24ae23e1966cfa86f79adb89c9f5985
parent b5bd82753296f765e6aa0ebb5d60c3e0a93cac3c
author ElenaVerstova <isip_e.v.verstova@mpt.ru> 1613849667 +0300
committer ElenaVerstova <isip_e.v.verstova@mpt.ru> 1613849667 +0300

Added index.html
```

Рисунок 80 - Просмотр объекта коммита



```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git type c2bcb76
commit

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git dump c2bcb76
tree 4e279a99e24ae23e1966cfa86f79adb89c9f5985
parent b5bd82753296f765e6aa0ebb5d60c3e0a93cac3c
author ElenaVerstova <isip_e.v.verstova@mpt.ru> 1613849667 +0300
committer ElenaVerstova <isip_e.v.verstova@mpt.ru> 1613849667 +0300

Added index.html

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |
```

Рисунок 81 - Использование алиасов

Для просмотра дерева каталогов необходимо использовать его хэш (рисунок 20).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git cat-file -p 4e279a9
100644 blob 1240583197c9a4507a2fb0d59eb1a82886844e57    a.html
100644 blob af60016690b87c1f1757db52ac1baf43ff3419a8    b.html
100644 blob 03ee26631ef36c173c9c1a9f7e51dd419f72482d    index.html
040000 tree 34c23e296c4e5b2207ddeb4a4aaf565abdf9e0d0    lib
100644 blob 1eded30425a18ead4e57682660b74c8645761b51    v.html

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |
```

Рисунок 82 - Просмотр дерева каталогов

Затем нужно просмотреть каталог lib (рисунок 21).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git cat-file -p 34c23e296
100644 blob 07978d6ac75191ca3ce7aaca139e95e951fe230b    hello.html

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$
```

Рисунок 83 - Просмотр каталога lib

И затем требуется вывести содержимое файла hello.html (рисунок 22).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git cat-file -p 07978d6a
<!-- Author: Elena Verstova (isip_e.v.verstova@mpt.ru) -->
<html>
  <head>

  </head>
  <body>
    <h1>Hello, world</h1>
    <h2>The cake is a lie.</h2>
  </body>
</html>
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$
```

Рисунок 84 - Вывод содержимого файла hello.html

Аналогичным образом можно просмотреть содержимое файла, каким оно было в самом первом коммите, как показано на рисунке 23. Для этого требуется использовать лишь нужные хэши.

```
Korra@Korra MINGW64 ~/hello (master)
$ git hist
* 23cfce1 2025-06-03 | Add an author comment (HEAD -> master) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase (style) [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]
```

Рисунок 85 - Просмотр содержимого файла при первом коммите

_УП_4

Работа с Git

Создание ветки

Для начала необходимо создать ветку style с помощью команды `git checkout -b style` (рисунок 1).

```
Korra@Korra MINGW64 ~/hello (master)
$ git checkout style
Switched to branch 'style'

Korra@Korra MINGW64 ~/hello (style)
$
```

Рисунок 86 - Создание новой ветки style

Затем нужно создать файл стилей (рисунок 2) и внести в него код, показанный на рисунке 3.

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ touch lib/style.css

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 87 - Создание файла стилей

```
# style.css
C: > Users > Korra > OneDrive > Desktop > hello > lib > # style.css > h1
1  h1 {
2      color: red;
3  }
```

Рисунок 88 - Код style.css

После этого надо произвести индексацию и коммит (рисунок 4).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Added css stylesheet"
[master 869858d] Added css stylesheet
1 file changed, 3 insertions(+)
create mode 100644 lib/style.css

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 89 - Индексация и коммит нового файла

Далее требуется изменить основную страницу hello.html и закоммитить изменения (рисунки 5-6).

```
C: > Users > Korra > OneDrive > Desktop > hello > hello.html > html > head
1  <!--Author: Rahimov Asad (abramovaleva3007@gmail.com)-->
2  <html>
3      <head>
4          <link rel="stylesheet" type="text/css" media="all" href="style.css">
5      </head>
6      <body>
7          <h1>Hello, world</h1>
8          <h2>The cake is a lie.</h2>
9      </body>
10 </html>
11
12
```

Рисунок 90 - Изменения в файле hello.html

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Hello.html uses style.css"
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:   index.html

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

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

Рисунок 91 - Индексация и коммит

Далее аналогичные действия нужно осуществить с файлом index.html, как это показано на рисунках 7-8.

```
C: > Users > Korra > OneDrive > Desktop > hello > <> index.html > html > html
1  <html>
2      <head>
3          <link rel="stylesheet" type="text/css" media="all" href="style.css">
4      </head>
5      <body>
6          <iframe src="lib/hello.html" width="200" height="200"></iframe>
7      </body>
8  </html>
9  |
```

Рисунок 92 - Изменения в файле index.html

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git add index.html

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Update index.html"
[master 4d3884a] Update index.html
1 file changed, 8 insertions(+), 4 deletions(-)

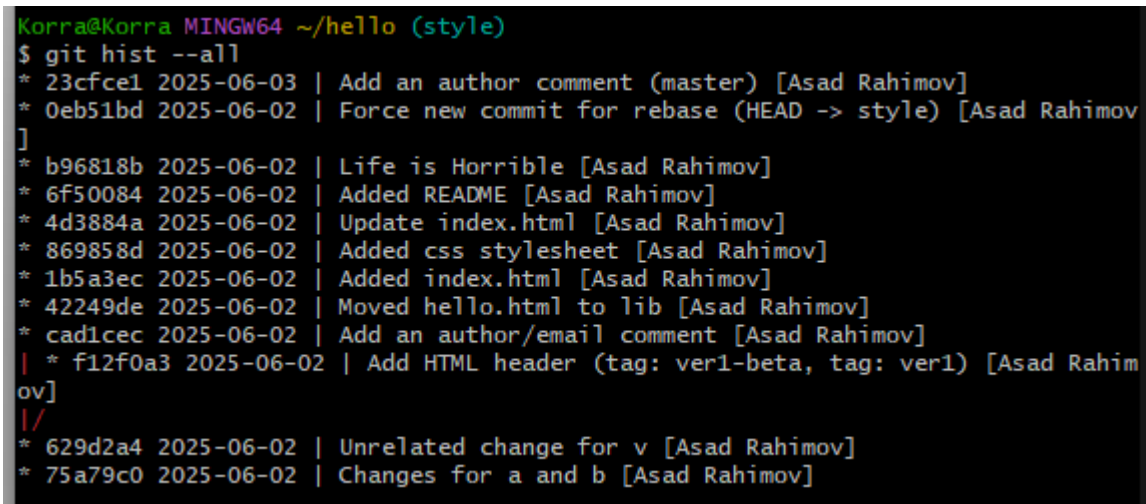
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 93 - Индексация и коммит

После выполнения предыдущих действий была создана новая ветка style с 3 коммитами.

[Навигация по веткам](#)

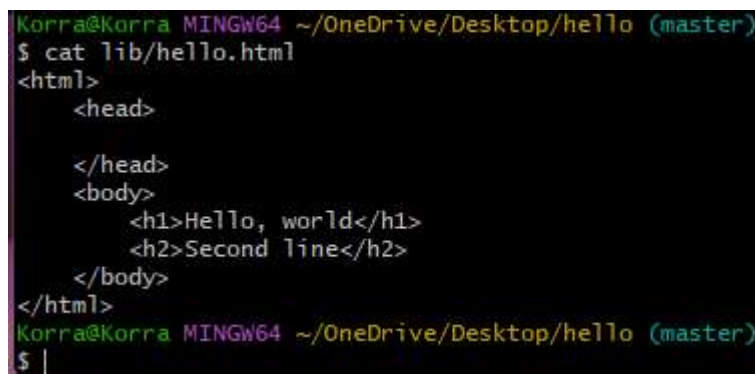
При просмотре истории, как на рисунке 9, можно увидеть, что теперь в проекте 2 ветки.



```
Korra@Korra MINGW64 ~/hello (style)
$ git hist --all
* 23cfce1 2025-06-03 | Add an author comment (master) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase (HEAD -> style) [Asad Rahimov]
]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]
```

Рисунок 94 - Просмотр истории

Для переключения на ветку master используется команда `git checkout master` (рисунок 10). После переключения на нужную ветку при выводе файла `hello.html` можно увидеть, что изменения отсутствуют (по причине того, что они закоммичены в другой ветке).



```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ cat lib/hello.html
<html>
  <head>

  </head>
  <body>
    <h1>Hello, world</h1>
    <h2>Second line</h2>
  </body>
</html>
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 95 - Переключение на ветку master

При переключении на ветку `style` файл `hello.html` будет иметь другое содержание (рисунок 11).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ cat lib/Hello.html
<!--Author: Rahimov Asad (abramovaleva3007@gmail.com)-->
<html>
  <head>
    <link rel="stylesheet" type="text/css" media="all" href="style.css" />
  </head>
  <body>
    <h1>Hello, world</h1>
    <h2>The cake is a lie.</h2>
  </body>
</html>

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$
```

Рисунок 96 - Переключение на ветку style

Изменения в ветке master

Необходимо переключиться на ветку master (рисунок 12) и добавить файл README (рисунки 13-14).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git checkout master
Switched to branch 'master'

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$
```

Рисунок 97 - Переключение на ветку master

```
C: > Users > Korra > OneDrive > Desktop > hello > ① README
1 This is the Hello World example from the Git tutorial |
```

Рисунок 98 - Содержимое файла README

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git add README

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Added README"
[master 6f50084] Added README
1 file changed, 1 insertion(+)
create mode 100644 README

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ |
```

Рисунок 99 - Индексация и коммит

Просмотр отличающихся веток

На рисунке 15 можно увидеть дерево коммитов.



```
Korra@Korra MINGW64 ~/hello (style)
$ git hist --all
* 23cfce1 2025-06-03 | Add an author comment (master) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase (HEAD -> style) [Asad Rahimov]
]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]
```

Рисунок 100 - Дерево коммитов

Слияние

Слияние переносит изменения из двух веток в одну. Для слияния нужно перейти на ветку style и с помощью команды `git merge master` совместить ветки (рисунки 16-18).



```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git checkout style
Switched to branch 'style'

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git merge master
Updating 4d3884a..6f50084
Fast-forward
 README | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 README

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$
```

Рисунок 101 - Переключение на ветку style

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (style)
$ git merge master
hint: Waiting for your editor to close the file...
(electron) Sending uncompressed crash reports is deprecated and will be removed
in a future version of Electron. Set { compress: true } to opt-in to the new beh
avior. Crash reports will be uploaded gzipped, which most crash reporting server
s support.
[main 2021-02-24T11:53:30.275Z] update#setState idle
(node:45768) electron: The default of contextIsolation is deprecated and will be
changing from false to true in a future release of Electron. See https://githu
b.com/electron/electron/issues/23506 for more information
(node:45768) electron: The default of contextIsolation is deprecated and will be
changing from false to true in a future release of Electron. See https://githu
b.com/electron/electron/issues/23506 for more information
[main 2021-02-24T11:54:00.303Z] update#setState checking for updates
[main 2021-02-24T11:54:00.790Z] update#setState idle
Merge made by the 'recursive' strategy.
 README | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 README
```

Рисунок 102 - Слияние с веткой master

Git

```
Korra@Korra MINGW64 ~/hello (style)
$ git hist --all
* 23cfce1 2025-06-03 | Add an author comment (HEAD -> style, master) [Asad Rahim
ov]
* 0eb51bd 2025-06-02 | Force new commit for rebase [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahim
ov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]
```

Рисунок 103 - Просмотр истории

Создание конфликта

Для того, чтобы создать конфликт необходимо перейти в ветку master и внести изменения в файл hello.html (рисунки 19-21).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git checkout master
Switched to branch 'master'

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$
```

Рисунок 104 - Переход в ветку master

```

<> Hello.html X
C: > Users > Korra > OneDrive > Desktop > hello > <> Hello.html > html > html
1  <!--Author: Rahimov Asad (abramovaleva3007@gmail.com)-->
2  <html>
3      <head>
4          <!--no style-->
5      </head>
6      <body>
7          <h1>Hello, world. Life is Horrible</h1>
8          <h2>The cake is a lie.</h2>
9      </body>
10 </html>
11 |

```

Рисунок 105 - Внесение изменений

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git commit -m "Life is Horrible"
[master b96818b] Life is Horrible
1 file changed, 9 insertions(+), 8 deletions(-)

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$

```

Рисунок 106 - Индексация и коммит

После выполнения предыдущих действий при просмотре веток можно будет увидеть конфликт в виде, как на рисунке 22.

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git log --all --pretty=format:"%h %ad | %s%d [%an]" --graph --date=short
* b96818b 2025-06-02 | Life is Horrible (HEAD -> master) [Asad Rahimov]
* 6f50084 2025-06-02 | Added README (style) [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

```

Рисунок 107 - Конфликт изменений

Разрешение конфликтов

При попытке объединить ветку style с master будет показана ошибка из-за конфликта, как показано на рисунке 23.

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git checkout style
Switched to branch 'style'

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git merge master
Updating 6f50084..b96818b
Fast-forward
 lib/hello.html | 17 ++++++-----
 1 file changed, 9 insertions(+), 8 deletions(-)

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ |

```

Рисунок 108 - Ошибка при слиянии

При открытии файла hello.html конфликт будет показан (рисунок 24).

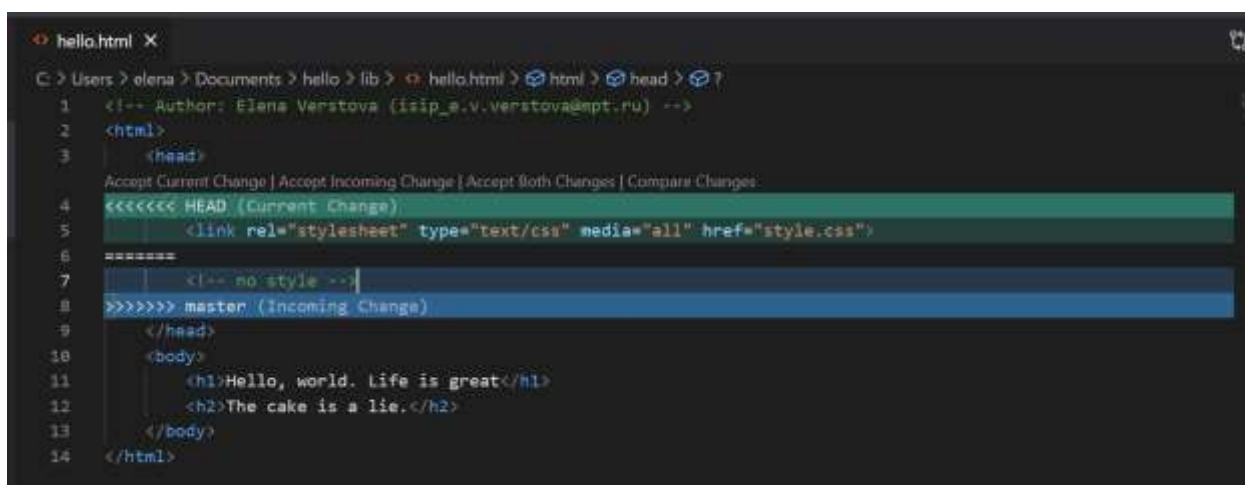


Рисунок 109 - Просмотр файла hello.html при наличии конфликта

Чтобы решить конфликт, нужно внести изменения вручную (рисунок 25).

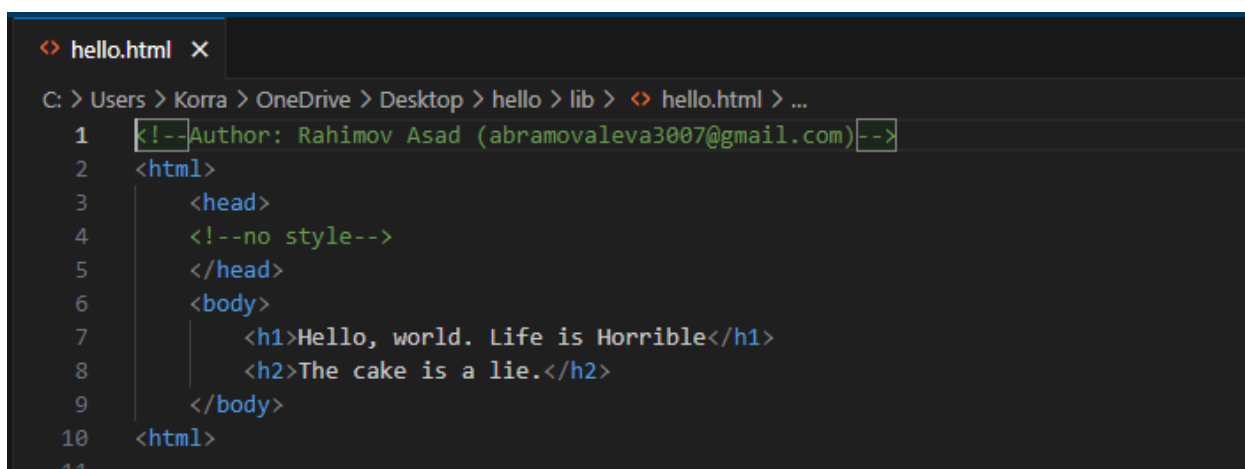


Рисунок 110 - Решение конфликта вручную

Затем следует произвести индексацию и коммит (рисунок 26).


```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (style|MERGING)
$ git add lib/hello.html

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (style|MERGING)
$ git commit -m "Merged master fixed conflict."
[style 945ce49] Merged master fixed conflict.
```

Рисунок 111 - Индексация и коммит

_УП_5

Работа с Git

Сброс ветки style

Для сброса ветки необходимо применить команду `reset --hard` до требуемой точки (рисунки 1-2).

```
Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git hist
* b96818b 2025-06-02 | Life is Horrible (HEAD -> style, master) [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$
```

Рисунок 112 - Просмотр истории

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git reset --hard
HEAD is now at b96818b Life is Horrible

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git hist --all
* b96818b 2025-06-02 | Life is Horrible (HEAD -> style, master) [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$

```

Рисунок 113 - Сброс ветки style

Сброс ветки master

Аналогичные действия нужно произвести и для ветки master (рисунки 3-4).

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git reset --hard
HEAD is now at b96818b Life is Horrible

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git hist --all
* b96818b 2025-06-02 | Life is Horrible (HEAD -> style, master) [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$

```

Рисунок 114 - Переключение на master и просмотр истории


```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git reset --hard
HEAD is now at b96818b Life is Horrible

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$ git hist --all
* b96818b 2025-06-02 | Life is Horrible (HEAD -> style, master) [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (style)
$

```

Рисунок 115 - Сброс ветки master

Перебазирование

Команду rebase можно использовать вместо команды merge (рисунок 5).

```

Korra@Korra MINGW64 ~/hello (style)
$ git rebase master
Current branch style is up to date.

Korra@Korra MINGW64 ~/hello (style)
$ git hist
* 23cfce1 2025-06-03 | Add an author comment (HEAD -> style, master) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/hello (style)

```

Рисунок 116 - Перебазирование веток

Слияние в ветку master

Далее требуется произвести слияние веток с помощью merge (рисунки 6-7).

```

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$ git merge style
Updating b96818b..0eb51bd
Fast-forward
 dummy_file.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 dummy_file.txt

Korra@Korra MINGW64 ~/OneDrive/Desktop/hello (master)
$

```

Рисунок 117 - Слияние веток

```

* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

Korra@Korra MINGW64 ~/hello (style)
$ git hist
* 23cfce1 2025-06-03 | Add an author comment (HEAD -> style, master) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]

```

Рисунок 118 - Просмотр истории

Клонирование репозитория

Далее требуется научиться делать копии репозитория. Для этого необходимо перейти в рабочий каталог и затем использовать команду `git clone`. Все данные действия показаны на рисунке 8.

```
Korra@Korra MINGW64 ~/Documents (style)
$ ls
cloned_hello/  hello/  'Мои видеозаписи'@  'Моя музыка'@  'мои рисунки'@
Korra@Korra MINGW64 ~/Documents (style)
$
```

```
Korra@Korra MINGW64 ~/Documents (style)
$ cd cloned_hello

Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ ls
README  a.html  b.html  dummy_file.txt  index.html  lib/  v.html
Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$
```

Рисунок 119 - Переход в рабочий каталог и его клонирование

Просмотр клонированного репозитория

После этого можно просмотреть клонированный репозиторий (рисунки 9-10).

```
elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$ cd cloned_hello

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ ls
README  a.html  b.html  index.html  lib/  v.html
elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ |
```

Рисунок 120 - Просмотр содержимого клонированного репозитория

```
Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ git hist --all
* 0eb51bd 2025-06-02 | Force new commit for rebase (HEAD -> master, origin/style, origin/master, origin/HEAD) [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
* f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahimov]
//
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]
Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ |
```

Рисунок 121 - Просмотр логов клонированного каталога

Origin

Origin – имя по умолчанию. Просмотр данных о нем возможен с помощью команд, показанных на рисунке 11.

```
Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ git remote
origin

Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ git remote show origin
* remote origin
  Fetch URL: C:/Users/Korra/Documents/hello
  Push URL: C:/Users/Korra/Documents/hello
  HEAD branch: master
  Remote branches:
    master tracked
    style tracked
  Local branch configured for 'git pull':
    master merges with remote master
  Local ref configured for 'git push':
    master pushes to master (up to date)

Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ |
```

Рисунок 122 - Просмотр данных об origin

Удаленные ветки

Для просмотра удаленных веток используется команда `git branch -a` (рисунок 12).

```
Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ git branch
* master

Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$ git branch -a
* master
remotes/origin/HEAD -> origin/master
remotes/origin/master
remotes/origin/style

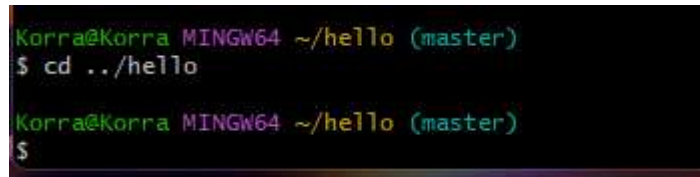
Korra@Korra MINGW64 ~/Documents/cloned_hello (master)
$
```

Рисунок 123 - Просмотр удаленных веток

УП_6

Работа с Git

Сначала необходимо внести изменения в оригинальный репозиторий. Для этого нужно перейти в данный репозиторий (рисунок 1).

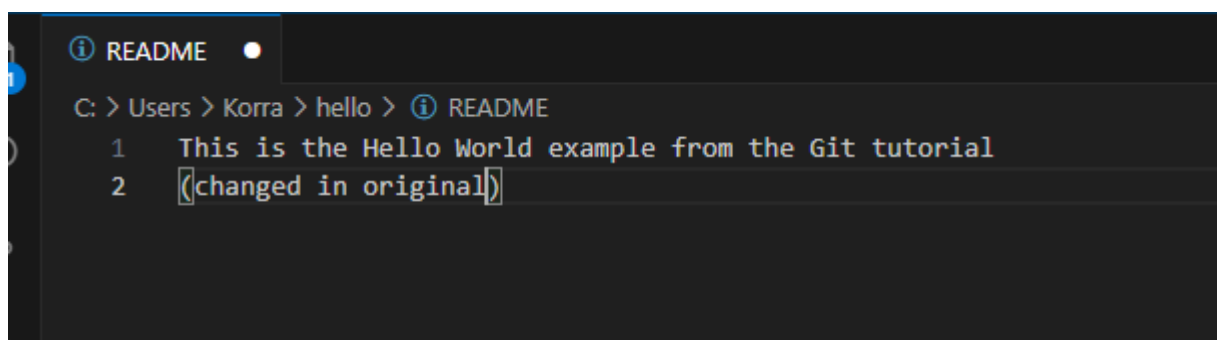


```
Korra@Korra MINGW64 ~/hello (master)
$ cd ../hello

Korra@Korra MINGW64 ~/hello (master)
$
```

Рисунок 124 - Переход в оригинальный репозиторий

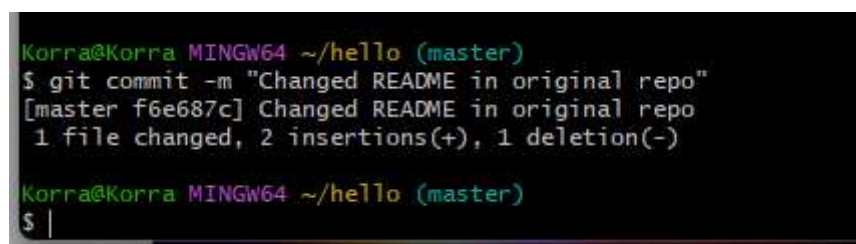
Далее надо внести изменения в файл README (рисунок 2) и затем произвести индексацию и коммит (рисунок 3).



The screenshot shows a code editor with a file named README. The content of the file is as follows:

```
C: > Users > Korra > hello > README
1 This is the Hello World example from the Git tutorial
2 [(changed in original)]
```

Рисунок 125 - Изменения в файле README



```
Korra@Korra MINGW64 ~/hello (master)
$ git commit -m "Changed README in original repo"
[master f6e687c] Changed README in original repo
1 file changed, 2 insertions(+), 1 deletion(-)

Korra@Korra MINGW64 ~/hello (master)
$ |
```

Рисунок 126 - Индексация и коммит новых изменений

Далее требуется перейти в клонированный репозиторий и извлечь изменения с помощью команды `git fetch` (рисунок 4) и просмотреть историю (рисунок 5).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ cd ../cloned_hello

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 344 bytes | 11.00 KiB/s, done.
From C:/Users/elena/Documents/hello
  9dadf96..5906087  master    -> origin/master
```

Рисунок 127 - Извлечение изменений

```
Korra@Korra MINGW64 ~/hello (master)
$ git hist --all
* f6e687c 2025-06-03 | Changed README in original repo (HEAD -> master) [Asad Ra
himov]
* 23cfce1 2025-06-03 | Add an author comment (style) [Asad Rahimov]
* 0eb51bd 2025-06-02 | Force new commit for rebase [Asad Rahimov]
* b96818b 2025-06-02 | Life is Horrible [Asad Rahimov]
* 6f50084 2025-06-02 | Added README [Asad Rahimov]
* 4d3884a 2025-06-02 | Update index.html [Asad Rahimov]
* 869858d 2025-06-02 | Added css stylesheet [Asad Rahimov]
* 1b5a3ec 2025-06-02 | Added index.html [Asad Rahimov]
* 42249de 2025-06-02 | Moved hello.html to lib [Asad Rahimov]
* cad1cec 2025-06-02 | Add an author/email comment [Asad Rahimov]
| * f12f0a3 2025-06-02 | Add HTML header (tag: ver1-beta, tag: ver1) [Asad Rahim
ov]
|/
* 629d2a4 2025-06-02 | Unrelated change for v [Asad Rahimov]
* 75a79c0 2025-06-02 | Changes for a and b [Asad Rahimov]
```

Рисунок 128 - Просмотр истории

При попытке вывести содержимое файла README можно увидеть, что изменения не были внесены (рисунок 6).

```
Korra@Korra MINGW64 ~/hello (master)
$ cat README
This is the Hello World example from the Git tutorial
```

Рисунок 129 - Вывод содержимого файла README

Далее нужно слить извлеченные изменения в ветку master (рисунок 7).


```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git merge origin/master
Updating 9dadf96..5906087
Fast-forward
 README | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ |
```

Рисунок 130 - Слияние изменений

И после выполнения предыдущего действия при выводе README можно будет увидеть последние изменения (рисунок 8).

```
Korra@Korra MINGW64 ~/hello (master)
$ cat README
This is the Hello World example from the Git tutorial
(changed in original)
Korra@Korra MINGW64 ~/hello (master)
```

Рисунок 131 - Вывод содержимого файла README

Также существует команда, объединяющая функции git fetch и git merge, которая показана на рисунке 9.

```
Korra@Korra MINGW64 ~/hello (master)
$ git pull
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=<remote>/<branch> master

Korra@Korra MINGW64 ~/hello (master)
$ |
```

Рисунок 132 - Команда git pull

Далее требуется добавить локальную ветку, которая будет отслеживать удаленную ветку (рисунок 10).


```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git branch --track style origin/style
Branch 'style' set up to track remote branch 'style' from 'origin'.

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git branch -a
* master
  style
remotes/origin/HEAD -> origin/master
remotes/origin/master
remotes/origin/style

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git hist --max-count=2
* 5906087 2021-03-10 | Changed README in original repo (HEAD -> master, origin/master, or
igin/HEAD) [ElenaVerstova]
* 9dadf96 2021-02-24 | Life is great! (origin/style, style) [ElenaVerstova]

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ |

```

Рисунок 133 - Добавление локальной ветки

Далее необходимо создать чистый репозиторий (рисунок 11).

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ cd ..

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$ git clone --bare hello hello.git
Cloning into bare repository 'hello.git'...
done.

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$ ls hello.git
HEAD config description hooks/ info/ objects/ packed-refs refs/

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$

```

Рисунок 134 - Создание чистого репозитория

Для добавления удаленного репозитория используется команда, показанная на рисунке 12.

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$ cd hello

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git remote add shared ../hello.git

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |

```

Рисунок 135 - Добавление удаленного репозитория

Затем требуется научиться отправлять изменения в удаленный репозиторий. Для этого сначала надо внести изменения, проиндексировать и произвести коммит (рисунок 13-14).



Рисунок 136 - Внесение изменений в файл

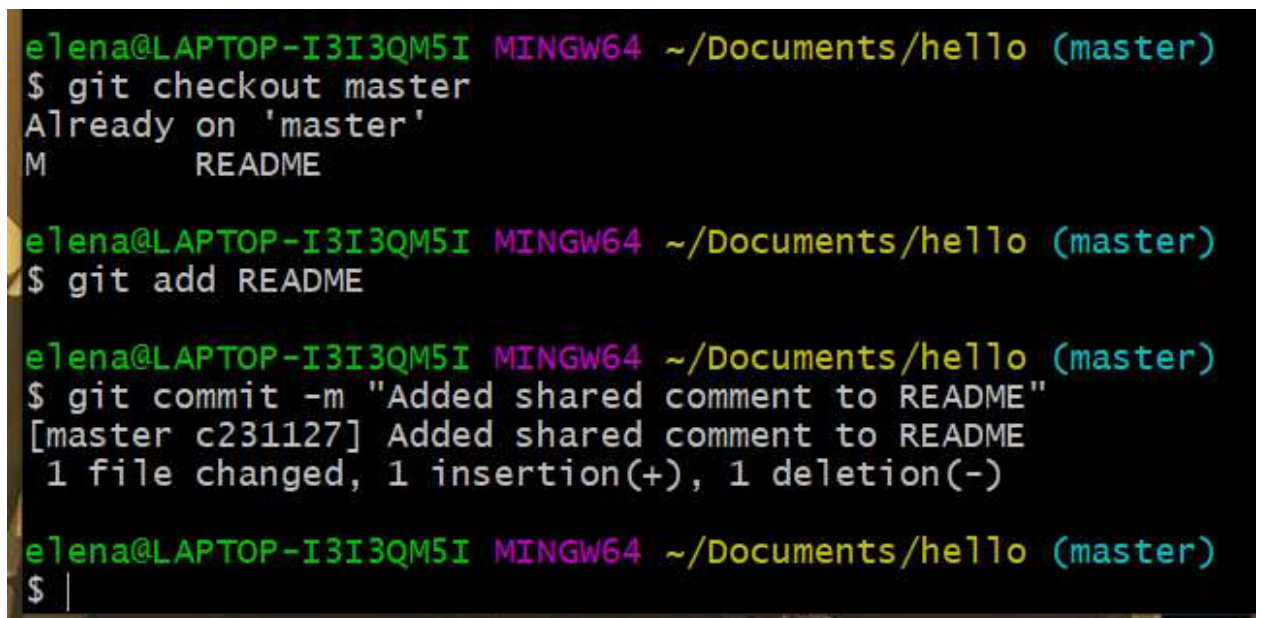
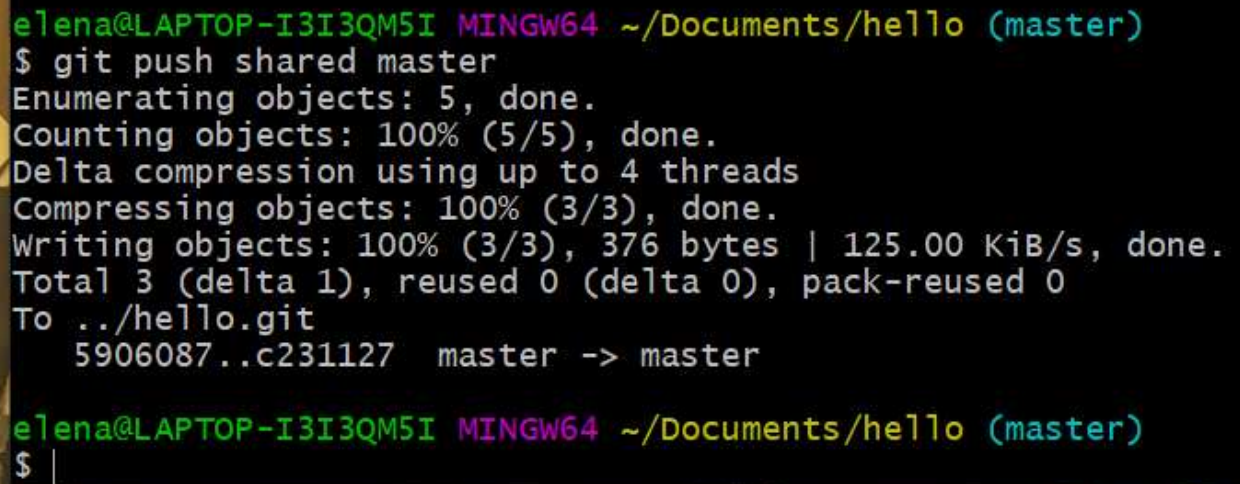


Рисунок 137 - Индексация и коммит

Далее надо отправить изменения в общий репозиторий, используя команду `git push shared master` (рисунок 15).

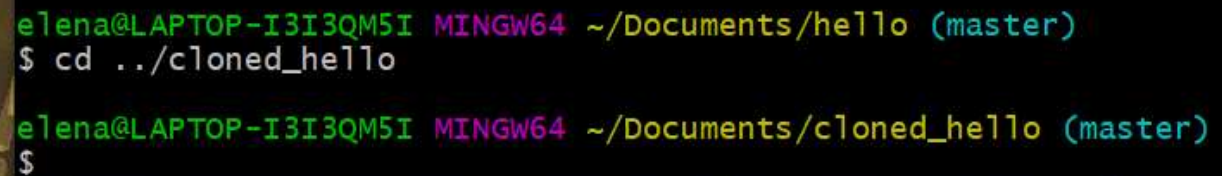


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git push shared master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 376 bytes | 125.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
To ../hello.git
    5906087..c231127  master -> master

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |
```

Рисунок 138 - Отправка изменений в общий репозиторий

Для извлечения общих изменений нужно перейти в клонированный каталог и выполнить перечень команд (рисунок 16).



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ cd ../cloned_hello

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$
```

Рисунок 139 - Переход в клонированный репозиторий


```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git remote add shared ../hello.git

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git branch --track shared master
Branch 'shared' set up to track local branch 'master'.

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ git pull shared master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 356 bytes | 11.00 KiB/s, done.
From ../hello
 * branch            master      -> FETCH_HEAD
 * [new branch]      master      -> shared/master
Updating 5906087..c231127
Fast-forward
 README | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)
$ cat README
This is the Hello World example from the Git tutorial
(Changed in the original and pushed to shared)
elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/cloned_hello (master)

```

Рисунок 140 - Команды, извлекающие общие изменения

Для настройки git сервера нужно выполнить команду, показанную на рисунке 18. Затем в другом окне можно проверить работу сервера, сделав копию проекта hello (рисунок 18).

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git daemon --verbose --export-all --base-path=.
[111720] Ready to rumble

```

Рисунок 141 - Настройка сервера

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git clone ~/Documents/hello.git network_hello
Cloning into 'network_hello'...
done.

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ cd network_hello

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello/network_hello (master)
$ ls
README  a.html  b.html  index.html  lib/  v.html

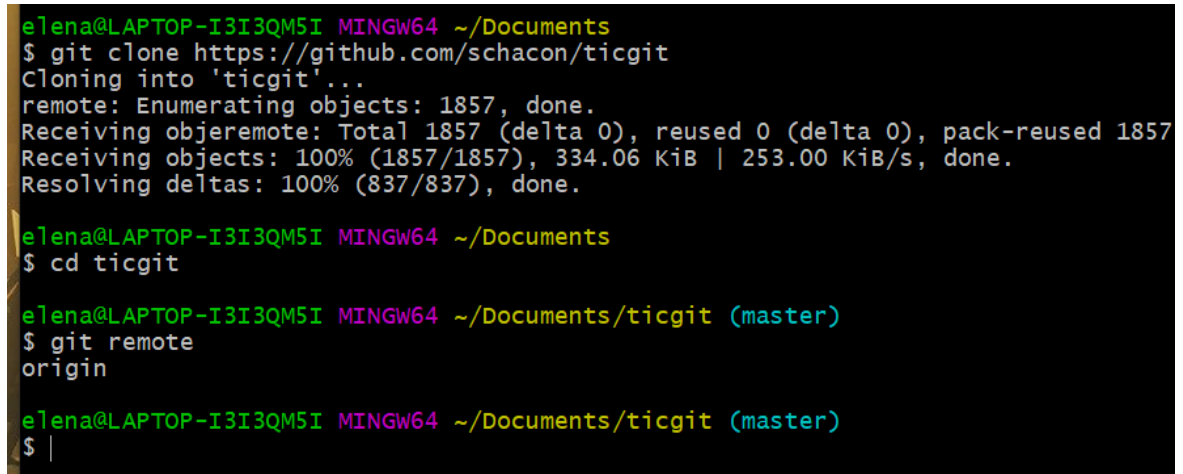
```

Рисунок 142 - Клонирование проекта

_УП_7

Работа с Git

Для того, чтобы просмотреть список настроенных удалённых репозиторий, необходимо запустить команду `git remote` (рисунок 1).



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$ git clone https://github.com/schacon/ticgit
Cloning into 'ticgit'...
remote: Enumerating objects: 1857, done.
remote: Total 1857 (delta 0), reused 0 (delta 0), pack-reused 1857
Receiving objects: 100% (1857/1857), 334.06 KiB | 253.00 KiB/s, done.
Resolving deltas: 100% (837/837), done.

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents
$ cd ticgit

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote
origin

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ |
```

Рисунок 143 – Клонирование репозитория и просмотр удаленных репозиторий

Можно указать ключ `-v`, чтобы просмотреть адреса для чтения и записи, привязанные к репозиторию (рисунок 2).



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote -v
origin https://github.com/schacon/ticgit (fetch)
origin https://github.com/schacon/ticgit (push)

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$
```

Рисунок 144 - Просмотр удаленных репозиторий с ключом `-v`

Для добавления удаленного репозитория с новым именем используется команда `git remote add` (рисунок 3).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote add pb https://github.com/paulboone/ticgit

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote -v
origin https://github.com/schacon/ticgit (fetch)
origin https://github.com/schacon/ticgit (push)
pb      https://github.com/paulboone/ticgit (fetch)
pb      https://github.com/paulboone/ticgit (push)
```

Рисунок 145 - Добавление удаленного репозитория

После задания имени репозиторию впоследствии его можно использовать вместо указания полного пути (рисунок 4).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git fetch pb
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Total 43 (delta 22), reused 22 (delta 22), pack-reused 21
Unpacking objects: 100% (43/43), 5.99 KiB | 16.00 KiB/s, done.
From https://github.com/paulboone/ticgit
 * [new branch]      master    -> pb/master
 * [new branch]      ticgit    -> pb/ticgit

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$
```

Рисунок 146 - Использование имени вместо пути

Для получения данных из удалённых проектов используется команда `git fetch` (рисунок 5).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git fetch ~/Documents/cloned_hello
remote: Enumerating objects: 39, done.
remote: Counting objects: 100% (39/39), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 39 (delta 12), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (39/39), 3.42 KiB | 10.00 KiB/s, done.
From C:/Users/elena/Documents/cloned_hello
 * branch            HEAD      -> FETCH_HEAD

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ |
```

Рисунок 147 - Получение данных из удаленных проектов

Для отправки изменений в удаленный репозиторий используется команда `git push` (рисунок 6).

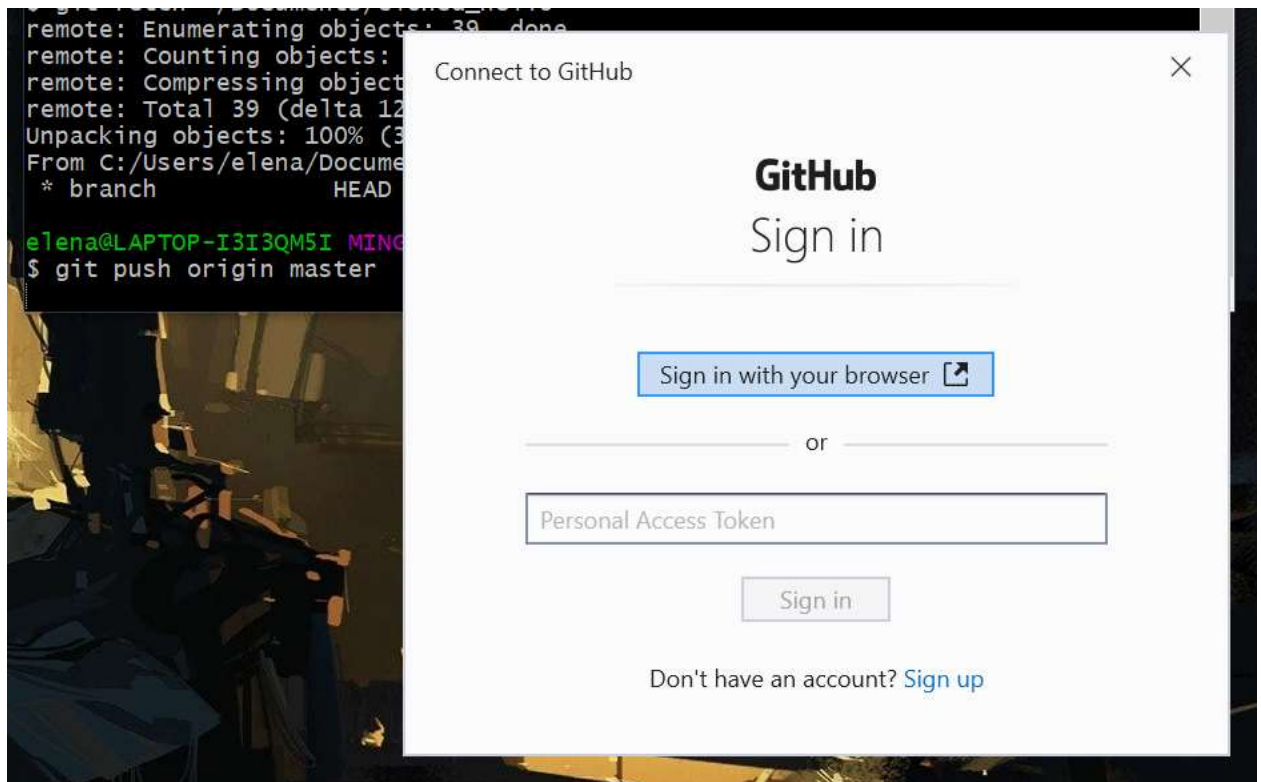


Рисунок 148 - Отправка изменений в удаленный репозиторий

Для получения информации об одном из удалённых репозиториях, можно использовать команду `git remote show` (рисунок 7).

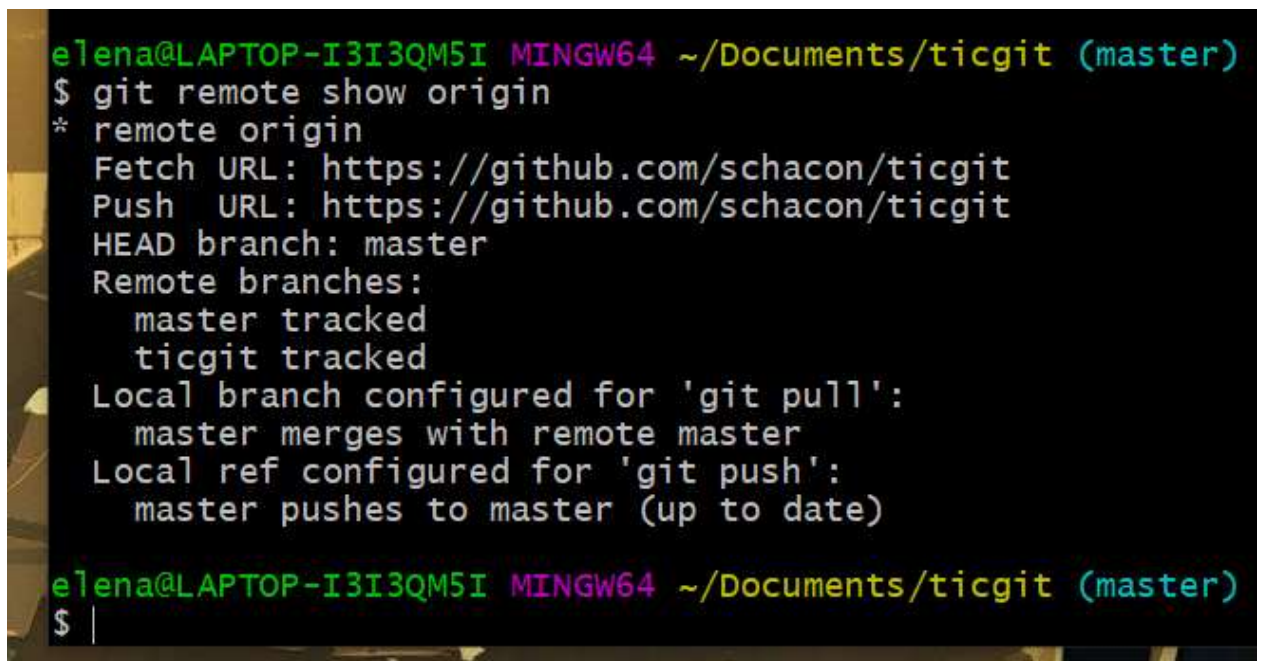


Рисунок 149 - Информация об удаленном репозитории

Для переименования удаленных репозиториях используется команда `git remote rename` (рисунок 8).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote rename pb paul

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote
origin
paul

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$
```

Рисунок 150 - Переименование удаленного репозитория

Для удаления удаленного репозитория нужно выполнить команду `git remote remove` (рисунок 9).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote remove paul

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ git remote
origin

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/ticgit (master)
$ |
```

Рисунок 151 - Удаление удаленного репозитория

Просмотреть существующие теги можно с помощью команды `git tag` (рисунок 10).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag
ver1
ver1-beta
ver2
```

Рисунок 152 - Просмотр тегов

Для создания аннотированной метки нужно выполнить команду, показанную на рисунке 11.

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag -a v2.1 -m "My version 2.1"

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag
v2.1
ver1
ver1-beta
ver2

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |
```

Рисунок 153 - Создание аннотированной метки

Команда `git show` осуществляет просмотр данных тегов вместе с коммитом (рисунок 12).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git show v2.1
tag v2.1
Tagger: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Fri Mar 19 11:18:30 2021 +0300

My version 2.1

commit c231127b40601b59292a5e3b0de3854e5341524c (HEAD -> master, tag: v2.1, shared/master)
Author: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Wed Mar 10 14:35:49 2021 +0300

    Added shared comment to README

diff --git a/README b/README
index 9091eb5..61fdad9 100644
--- a/README
+++ b/README
@@ -1,2 +1,2 @@
 This is the Hello World example from the Git tutorial
 -(changed in original)
 \ No newline at end of file
 +(Changed in the original and pushed to shared)
 \ No newline at end of file
 :...skipping...
tag v2.1
Tagger: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Fri Mar 19 11:18:30 2021 +0300
```

Рисунок 154 - Просмотр данных тега

Для создания легковесной метки не нужно передавать опции `-a`, `-s` и `-m`, надо указать только название (рисунок 13). Просмотр данных такой метки осуществляется также с помощью `git show` (рисунок 14).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag v2.1-lw

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag
v2.1
v2.1-lw
ver1
ver1-beta
ver2

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |
```

Рисунок 155 - Создание легковесной метки

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git show v2.1-lw
commit c231127b40601b59292a5e3b0de3854e5341524c (HEAD -> master, tag: v2.1-lw, tag: v2.1, shared/master)
Author: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Wed Mar 10 14:35:49 2021 +0300

    Added shared comment to README

diff --git a/README b/README
index 9091eb5..61fdad9 100644
--- a/README
+++ b/README
@@ -1,2 +1,2 @@
-This is the Hello World example from the Git tutorial
-(changed in original)
\ No newline at end of file
+(Changed in the original and pushed to shared)
:...skipping...
commit c231127b40601b59292a5e3b0de3854e5341524c (HEAD -> master, tag: v2.1-lw, tag: v2.1, shared/master)
Author: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Wed Mar 10 14:35:49 2021 +0300

    Added shared comment to README

diff --git a/README b/README
index 9091eb5..61fdad9 100644
```

Рисунок 156 - Просмотр данных тега

Для отметки определенного коммита тегом надо указать его хэш (рисунки 15-17).


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git log --pretty=oneline
c231127b40601b59292a5e3b0de3854e5341524c (HEAD -> master, tag: v2.1-lw, tag: v2.1, shared/master) Added shared comment to README
5906087d2ffabe869dd34d6c5b2ec9ac096b6731 Changed README in original repo
9dadf96814ea7867a7271cb13882cc5b8bf062fe (style) Life is great!
3559bed39f209c82d41ba9a97363d292e18c5342 Added README
c2bcb764686a36610deccd4090f0e124c2fdb0e4 (tag: ver2) Added index.html
b5bd82753296f765e6aa0ebb5d60c3e0a93cac3c Moved hello.html to lib
aefcfd7f3a22e33c57a2925359205fa53a32b5b3 Add an author/email comment
d243b6b28cd79a79b0d27214ef82f2032d3de69c (tag: ver1) Added standard HTML tags
8de3e8034c7ba4cc85a9ae26844a5c51be5f5e66 (tag: ver1-beta) Added tag h2
ce52614162e1ea0e0ba63049b2c063a56f2d20a6 Unrelated change for v
e1f22769d657733ef4db0e7dd536a62a901a3395 Changes for a and b
f302a0ccb31ec3bd823f0bc11151e14317ccaef8 First Commit

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |
```

Рисунок 157 - Просмотр истории

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag -a ver3 5906087
hint: Waiting for your editor to close the file...
(electron) Sending uncompressed crash reports is deprecated and will be removed
in a future version of Electron. Set { compress: true } to opt-in to the new beh
avior. Crash reports will be uploaded gzipped, which most crash reporting server
s support.
[main 2021-03-19T08:22:54.139Z] update#setState idle
(node:28676) electron: The default of contextIsolation is deprecated and will be
changing from false to true in a future release of Electron. See https://githu
b.com/electron/electron/issues/23506 for more information
(node:28676) electron: The default of contextIsolation is deprecated and will be
changing from false to true in a future release of Electron. See https://githu
b.com/electron/electron/issues/23506 for more information
```

Рисунок 158 - Создание тега определенному коммиту

```
≡ TAG_EDITMSG ●
C: > Users > elena > Documents > hello > .git > ≡ TAG_EDITMSG
1 Tag message1|
2 #
3 # Write a message for tag:
4 # ver3
5 # Lines starting with '#' will be ignored.
6
```

Рисунок 159 - Ввод сообщения в текстовом редакторе

Данные этого тега можно просмотреть аналогичным образом (рисунок 18).

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git show ver3
tag ver3
Tagger: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Fri Mar 19 11:22:53 2021 +0300

Tag message1

commit 5906087d2ffabe869dd34d6c5b2ec9ac096b6731 (tag: ver3)
Author: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date:   Wed Mar 10 14:24:40 2021 +0300

    Changed README in original repo

diff --git a/README b/README
index 3f16465..9091eb5 100644
--- a/README
+++ b/README
@@ -1,2 @@
-This is the Hello World example from the Git tutorial
\ No newline at end of file
+This is the Hello World example from the Git tutorial
+(changed in original)
\ No newline at end of file

```

Рисунок 160 - Просмотр данных тега

По умолчанию, команда `git push` не отправляет теги на удалённые сервера. Нужно выполнить команду `git push shared` (рисунок 19).

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git push shared v2.1
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 170 bytes | 34.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To ../hello.git
 * [new tag]          v2.1 -> v2.1

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$

```

Рисунок 161 - Отправка тега на удаленный сервер

Можно использовать опцию `--tags` для команды `git push`. В таком случае все теги отправятся на удалённый сервер (рисунок 20).


```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git push shared --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 169 bytes | 42.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To ../hello.git
* [new tag]          v2.1-lw -> v2.1-lw
* [new tag]          ver3 -> ver3

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ |

```

Рисунок 162 - Отправка всех тегов на сервер

Для того, чтобы удалить тег, надо использовать команду `git tag` с параметром `-d` (рисунок 21).

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git tag -d v2.1-lw
Deleted tag 'v2.1-lw' (was c231127)

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$

```

Рисунок 163 - Удаление тега

Для удаления тега с сервера используется команда, показанная на рисунке 22.

```

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git push shared :refs/tags/v2.1-lw
To ../hello.git
- [deleted]          v2.1-lw

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$

```

Рисунок 164 - Удаление тегов с сервера

Для того, чтобы получить версии файлов, на которые указывает тег, можно выполнить `git checkout` для тега. Однако, это переведёт репозиторий в состояние «detached HEAD». Если в состоянии «detached HEAD» внести изменения и сделать коммит, то тег не изменится, при этом новый коммит не будет относиться ни к какой из веток, а доступ к нему можно будет получить

только по его хэшу. Поэтому в таком случае следует создать новую ветку (рисунки 23-24).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git checkout v2.1
Note: switching to 'v2.1'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

    git switch -

Turn off this advice by setting config variable advice.detachedHead to false
HEAD is now at c231127 Added shared comment to README
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello ((v2.1))
$ |
```

Рисунок 165 - Переключение на метку

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello ((v2.1))
$ git checkout -b version2 v2.1
Switched to a new branch 'version2'

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ |
```

Рисунок 166 - Создание новой ветки

Можно создать псевдонимы (алиасы) для команд. Создание алиасов и примеры их использования показаны на рисунках 25-30.

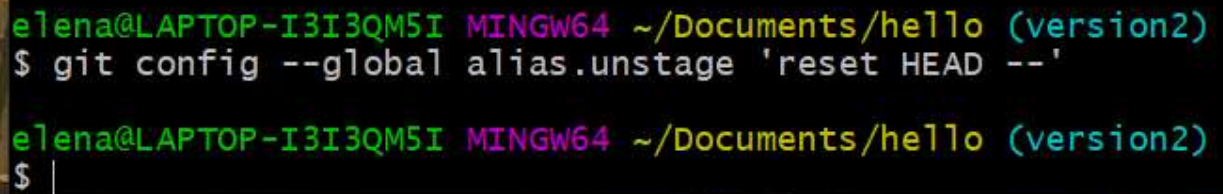
```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.co checkout

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.br branch

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.ci commit

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.st status
```

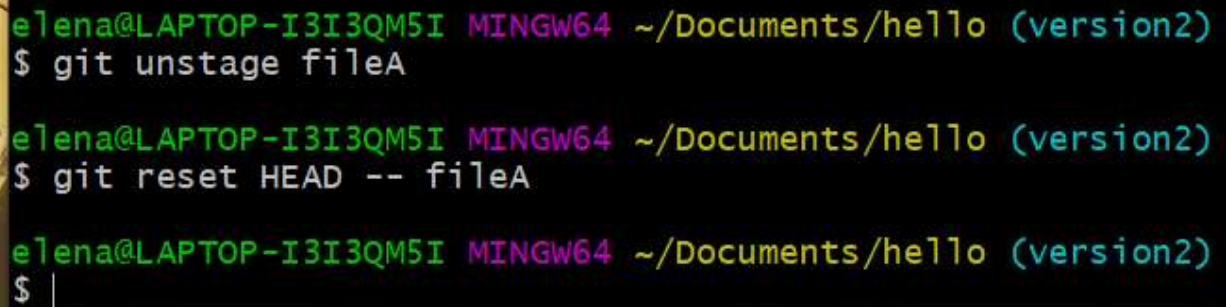
Рисунок 167 - Задание алиасов



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.unstage 'reset HEAD --'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ |
```

Рисунок 168 - Создание псевдонима исключения файла из индекса

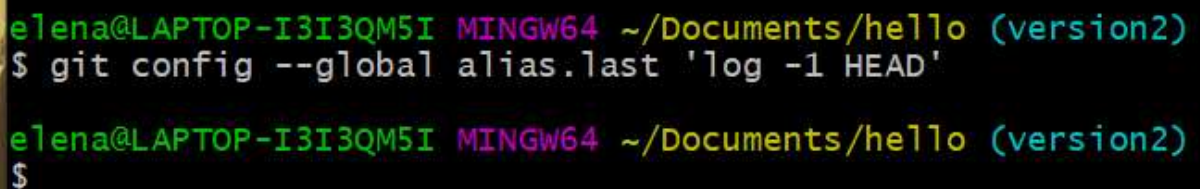


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git unstage fileA

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git reset HEAD -- fileA

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ |
```

Рисунок 169 - Использование созданного псевдонима



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.last 'log -1 HEAD'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$
```

Рисунок 170 - Создание алиаса для просмотра последнего коммита

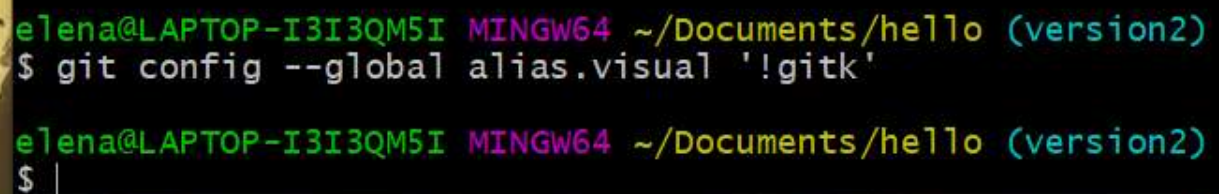


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git last
commit c231127b40601b59292a5e3b0de3854e5341524c (HEAD -> version2, tag: v2.1, shared/master, master)
Author: ElenaVerstova <isip_e.v.verstova@mpt.ru>
Date: Wed Mar 10 14:35:49 2021 +0300

    Added shared comment to README

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ |
```

Рисунок 171 - Результат работы созданного алиаса



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ git config --global alias.visual '!gitk'

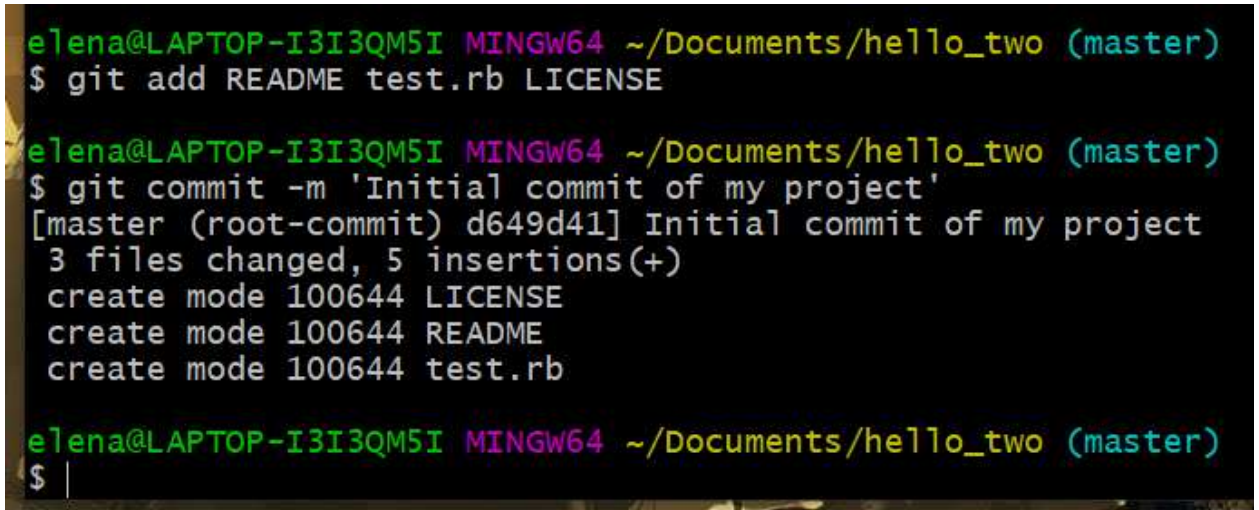
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (version2)
$ |
```

Рисунок 172 - Создание псевдонима внешней команды

_УП_8

Работа с Git

Для начала следует создать репозиторий, создать 3 файла и добавить их в коммит (рисунок 1).



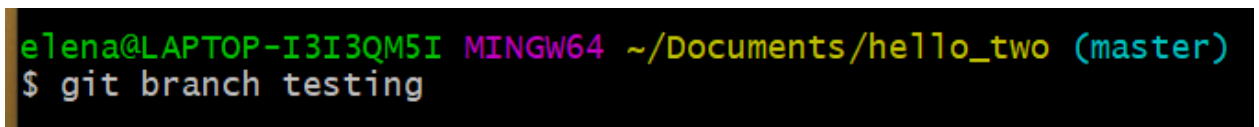
```
e|ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git add README test.rb LICENSE

e|ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git commit -m 'Initial commit of my project'
[master (root-commit) d649d41] Initial commit of my project
3 files changed, 5 insertions(+)
create mode 100644 LICENSE
create mode 100644 README
create mode 100644 test.rb

e|ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ |
```

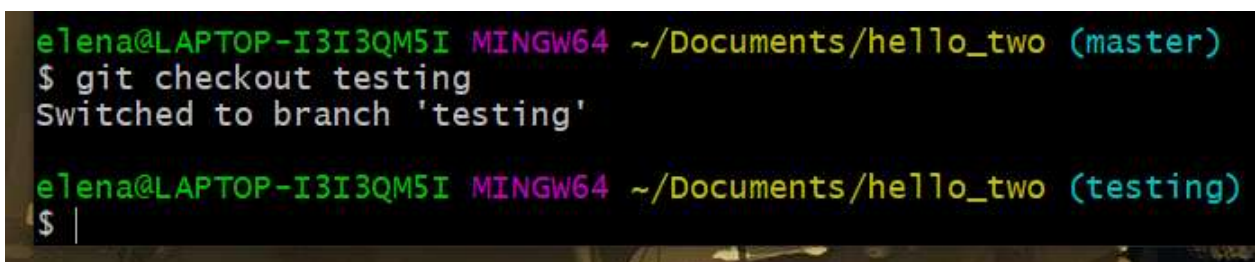
Рисунок 1 - Индексация и коммит 3 файлов

Затем надо создать ветку testing и переключиться на нее (рисунки 2-3).



```
e|ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch testing
```

Рисунок 2 - Создание ветки testing

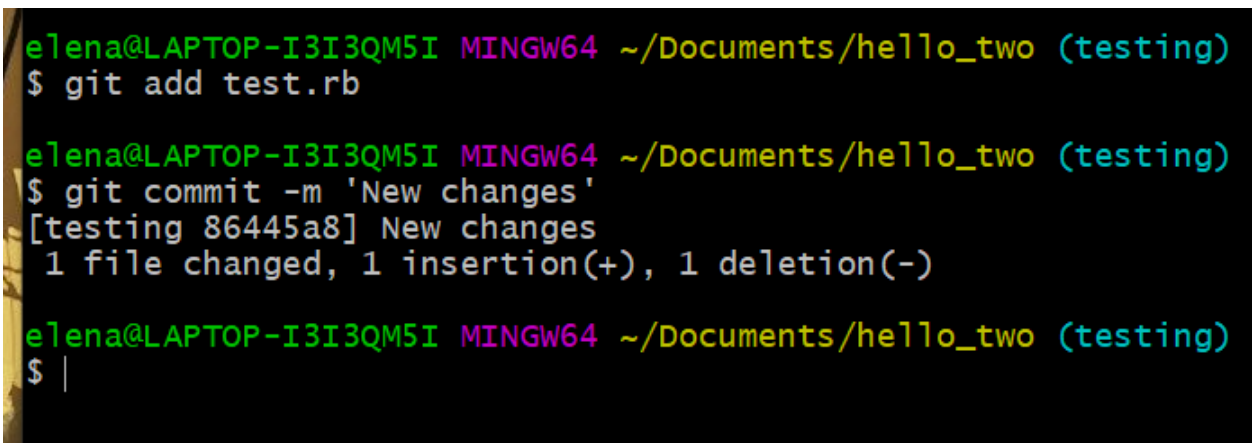


```
e|ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git checkout testing
Switched to branch 'testing'

e|ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (testing)
$ |
```

Рисунок 3 - Переключение на ветку testing

Далее надо внести изменения в файл test.rb и создать коммит (рисунок 4).




```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (testing)
$ git add test.rb

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (testing)
$ git commit -m 'New changes'
[testing 86445a8] New changes
1 file changed, 1 insertion(+), 1 deletion(-)

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (testing)
$ |
```

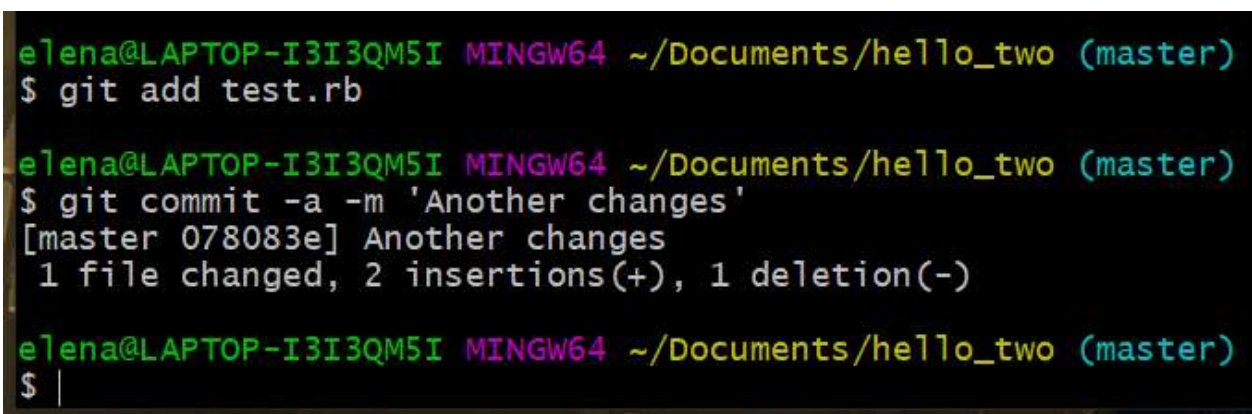
Рисунок 4 - Индексация и коммит файла test.rb

Затем необходимо переключиться на ветку master и внести изменения в файл test.rb на этой ветке (рисунки 5-6).



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (testing)
$ git checkout master
Switched to branch 'master'
```

Рисунок 5 - Переключение на ветку master



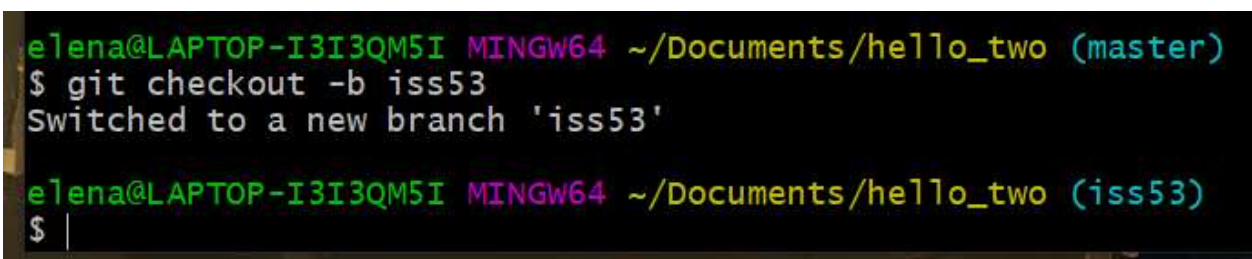
```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git add test.rb

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git commit -a -m 'Another changes'
[master 078083e] Another changes
1 file changed, 2 insertions(+), 1 deletion(-)

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ |
```

Рисунок 6 - Еще индексация и коммит test.rb

Команда git checkout -b позволяет сразу создать и переключиться на ветку (рисунок 7).

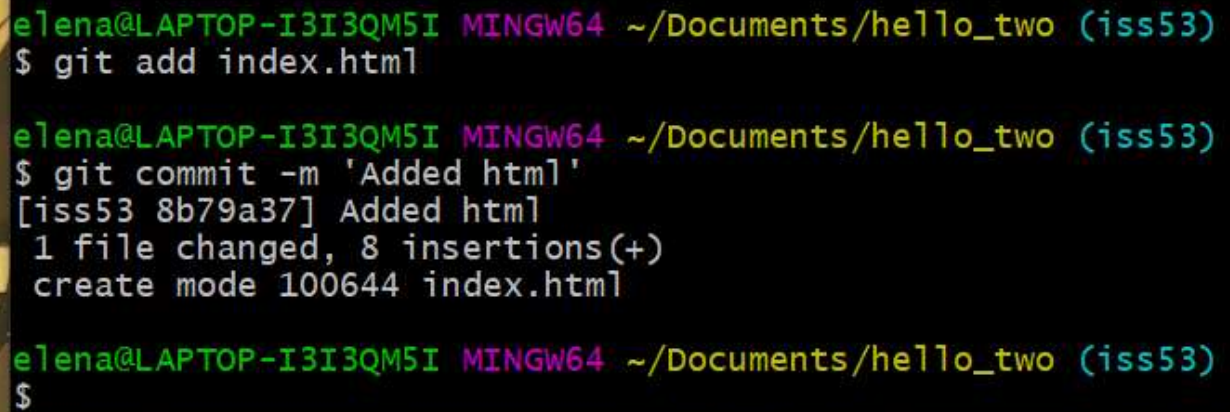


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git checkout -b iss53
Switched to a new branch 'iss53'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ |
```

Рисунок 7 - Создание и переключение на ветку iss53

В новой ветке нужно внести в файл изменения (рисунок 8).



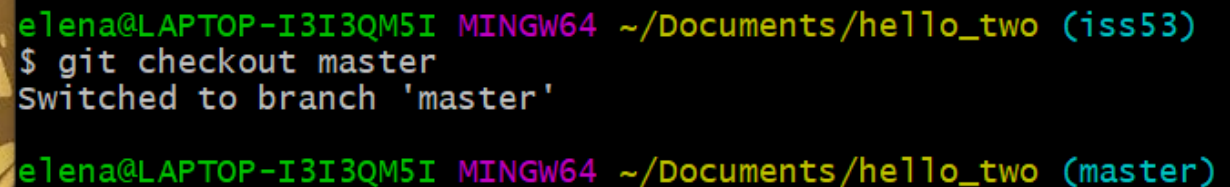
```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ git add index.html

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ git commit -m 'Added html'
[iss53 8b79a37] Added html
1 file changed, 8 insertions(+)
create mode 100644 index.html

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$
```

Рисунок 8 - Индексация и коммит файла index.html

Далее нужно переключить ветку на master (рисунок 9).



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ git checkout master
Switched to branch 'master'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
```

Рисунок 9 - Переключение на ветку master

Затем надо на ветке hotfix добавить изменения в файл index.html, а затем слить эту ветку и master (рисунки 10-11).



```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (hotfix)
$ git add index.html

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (hotfix)
$ git commit -a -m 'Fixed the broken email address'
[hotfix a935cd7] Fixed the broken email address
1 file changed, 8 insertions(+)
create mode 100644 index.html

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (hotfix)
$ |
```

Рисунок 10 - Индексация и коммит index.html на ветке hotfix


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (hotfix)
$ git checkout master
Switched to branch 'master'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git merge hotfix
Updating 078083e..a935cd7
Fast-forward
 index.html | 8 ++++++++
 1 file changed, 8 insertions(+)
 create mode 100644 index.html

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$
```

Рисунок 11 - Переключение на ветку master и объединение с веткой hotfix

После слияния ветку hotfix можно удалить (рисунок 12).1

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch -d hotfix
Deleted branch hotfix (was a935cd7).

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ |
```

Рисунок 12 - Удаление ветки hotfix

Затем требуется внести изменения в iss53, переключиться на master и слить эти ветки (рисунки 13-14).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ git add index.html

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ git commit -a -m 'Finished the new footer'
[iss53 7483233] Finished the new footer
 1 file changed, 3 insertions(+)

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ |
```

Рисунок 13 - Индексация и коммит index.html на ветке iss53

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (iss53)
$ git checkout master
Switched to branch 'master'

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git merge iss53
Updating 2f780f6..16ea1b3
Fast-forward
 index.html | 1 +
 1 file changed, 1 insertion(+)
```

Рисунок 14 - Слияние веток

После этого ветку iss53 нужно удалить (рисунок 15).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch -d iss53
Deleted branch iss53 (was 97393ba).

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$
```

Рисунок 15 - Удаление ветки iss53

_УП_9

Работа с Git

Команда `git branch` делает несколько больше, чем просто создаёт и удаляет ветки. При запуске без параметров, можно получить простой список имеющихся веток (рисунок 1). Символ `*`, стоящий перед веткой `master` указывает на ветку, на которую указывает HEAD).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch
* master
  newbranch
  testing

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ |
```

Рисунок 173 - Список существующих веток

Чтобы посмотреть последний коммит на каждой из веток, необходимо выполнить команду `git branch -v` (рисунок 2).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch -v
* master      c26e2f0 New CHanges
  newbranch   c26e2f0 New CHanges
  testing     0c5465f CHanges

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$
```

Рисунок 174 - Список веток с последними коммитами

Опции `--merged` и `--no-merged` могут отфильтровать этот список для вывода только тех веток, которые слиты или ещё не слиты в текущую ветку (рисунки 3-4).

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch --merged
* master
  newbranch
  testing
```

Рисунок 175 - Список веток слитых с текущей

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch --no-merged
```

Рисунок 176 - Список веток не слитых с текущей

Затем следует удалить ветку `testing` (рисунок 5). При наличии ошибок для удаления можно использовать параметр `-D`.

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch --d testing
Deleted branch testing (was 0c5465f).

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$
```

Рисунок 177 - Удаление ветки

Для получения списка удалённых веток и дополнительной информации используется команда `git remote show` (рисунок 6).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two/repo1 (main)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/Eternity-blip/repo1
  Push URL: https://github.com/Eternity-blip/repo1
  HEAD branch: main
  Remote branch:
    main tracked
  Local branches configured for 'git pull':
    main merges with remote main
    serv merges with remote main
  Local ref configured for 'git push':
    main pushes to main (local out of date)
```

Рисунок 178 - Просмотр удаленных веток

Для отправления изменений на удалённый сервер используется команда `git push <remote> <branch>` (рисунок 7).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two/repo1 (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 285 bytes | 95.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Eternity-blip/repo1
  6cdd680..f6b94bc  main -> main
```

Рисунок 179 - Отправка изменений

Далее при получении обновлений с сервера будет показана ссылка на удаленную ветку (рисунок 8).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two/repo1 (main)
$ git fetch origin
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 693 bytes | 18.00 KiB/s, done.
From https://github.com/Eternity-blip/repo1
  f6b94bc..68cbf5f  main      -> origin/main
```

Рисунок 180 - Выполнение команды git fetch

При необходимости можно создать локальную ветку на основе удаленной (рисунок 9).

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two/repo1 (main)
$ git checkout -b serv origin/main
Switched to a new branch 'serv'
Branch 'serv' set up to track remote branch 'main' from 'origin'.
```

Рисунок 181 - Создание ветки на основе удаленной ветки

Для удаления веток на удаленном сервере используется команда, показанная на рисунке 10.

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two/repo1 (serv)
$ git push origin :serv
To https://github.com/Eternity-blip/repo1
- [deleted]          serv
```

Рисунок 182 - Удаление ветки на сервере

Простой способ выполнить слияние двух веток – это команда `merge`. Другой способ – использование команды `rebase`, что означает перебазирование (рисунок 11). Это работает следующим образом: берётся общий родительский снимок двух веток (текущей, и той, поверх которой вы выполняете перебазирование), определяется дельта каждого коммита текущей ветки и сохраняется во временный файл, текущая ветка устанавливается на последний коммит ветки, поверх которой выполняется перебазирование, а затем по очереди применяются дельты из временных файлов.

Далее после этого надо переключиться на ветку `master` и выполнить перемотку.

```
e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git checkout exp
Switched to branch 'exp'

e1ena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (exp)
$ git rebase master
Successfully rebased and updated refs/heads/exp.
```

Рисунок 183 - Перемещение изменений

При наличии ответвления от ветки (сначала было ответвление на ветку `se`, а затем от нее на ветку `cl`), чтобы переместить изменения можно осуществить действия, показанные на рисунках 12-15.


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git rebase --onto master se cl
Successfully rebased and updated refs/heads/cl.
```

Рисунок 184 - Перемещение изменений с параметром onto

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (cl)
$ git checkout master
Switched to branch 'master'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git merge cl
Updating fafe213..08d256f
Fast-forward
 w.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 w.txt
```

Рисунок 185 - Слияние веток master и cl

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git rebase master se
Successfully rebased and updated refs/heads/se.
```

Рисунок 186 - Перемещение изменений

```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (se)
$ git checkout master
Switched to branch 'master'

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git merge se
Updating 08d256f..4a98db2
Fast-forward
 e.txt | 0
 q.txt | 0
 repo1 | 1 +
 3 files changed, 1 insertion(+)
 create mode 100644 e.txt
 create mode 100644 q.txt
 create mode 160000 repo1
```

Рисунок 187 - Слияние веток master и se

После этого перемещение будет осуществлено и ветки можно удалить (рисунок 16).


```
eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch -d cl
Deleted branch cl (was 08d256f).

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$ git branch -d se
Deleted branch se (was 4a98db2).

eIena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello_two (master)
$
```

Рисунок 188 - Удаление веток cl и se