

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.

```
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$ config --global user.name "Leskina Marina"  
bash: config: command not found  
  
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$ git config --global user.name "Leskina Marina"  
  
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$ git config --global user.email "linesssli998@gmail.com"  
  
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$
```

Рисунок 1 - Установка имени и электронной почты

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$ git config --global core.autocrlf true  
  
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$ git config --global core.safecrlf true  
  
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$
```

Рисунок 2 - Параметры окончаний строк

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$ git config --global core.quotePath off  
  
Admin@DESKTOP-F7T4IBU MINGW64 ~  
$
```

Рисунок 3 - Установка отображения Unicode

Создание проекта

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



Рисунок 4 - Создание каталога и файла

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

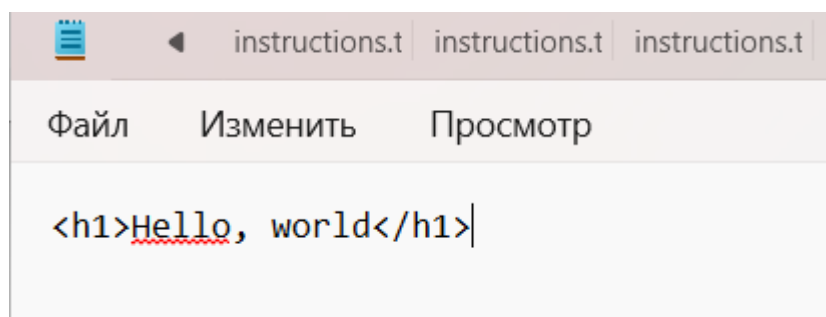


Рисунок 5 - Содержание файла

Для создания репозитория используется команда `git init` (рисунок 6).

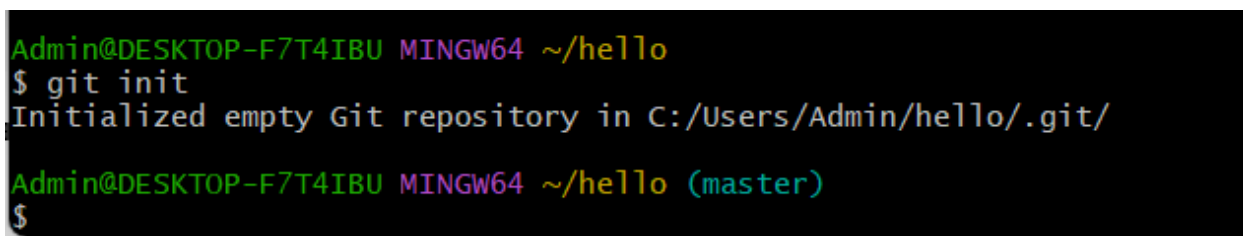


Рисунок 6 - Создание репозитория

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "first Commit"
[master (root-commit) 4791ef9] first Commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 7 - Добавление в репозиторий

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.

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/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).

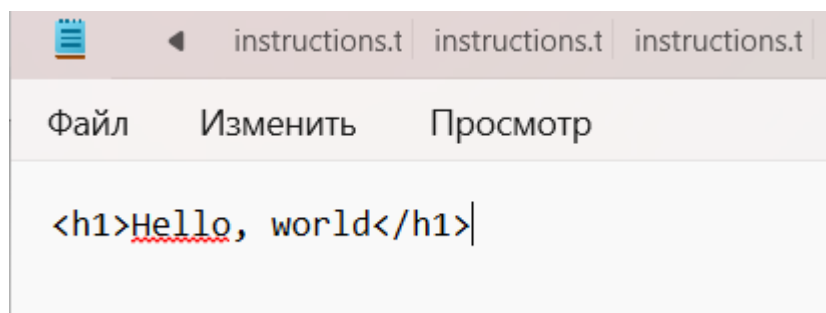


Рисунок 9 - Внесение изменений в файл

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/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")
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 10 - Сообщение о незафиксированных изменениях

Индексация изменений

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

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

```
$ git add b.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Changes for a and b"
[master 48389a7] Changes for a and b
3 files changed, 1 insertion(+)
create mode 100644 a.html
create mode 100644 b.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 13 - Индексация и коммит для 2 файлов

```
create mode 100644 b.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add v.html

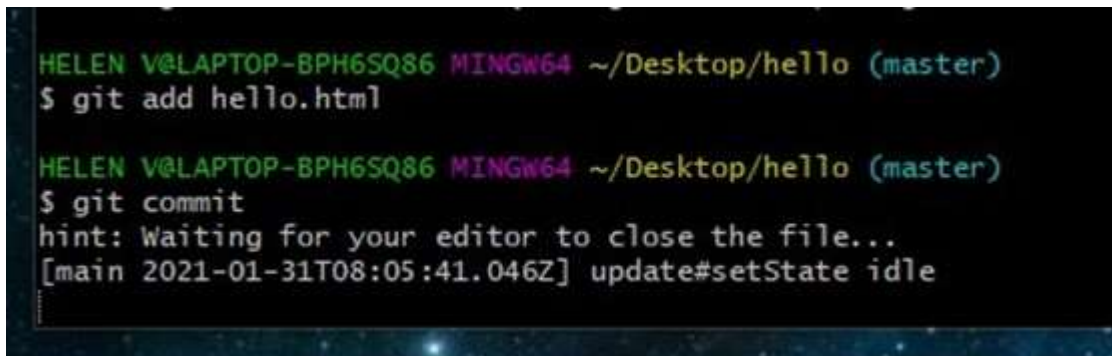
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Changes for v"
[master 22de71d] Changes for v
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 v.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 14 - Индексация и коммит для третьего файла

Коммит изменений

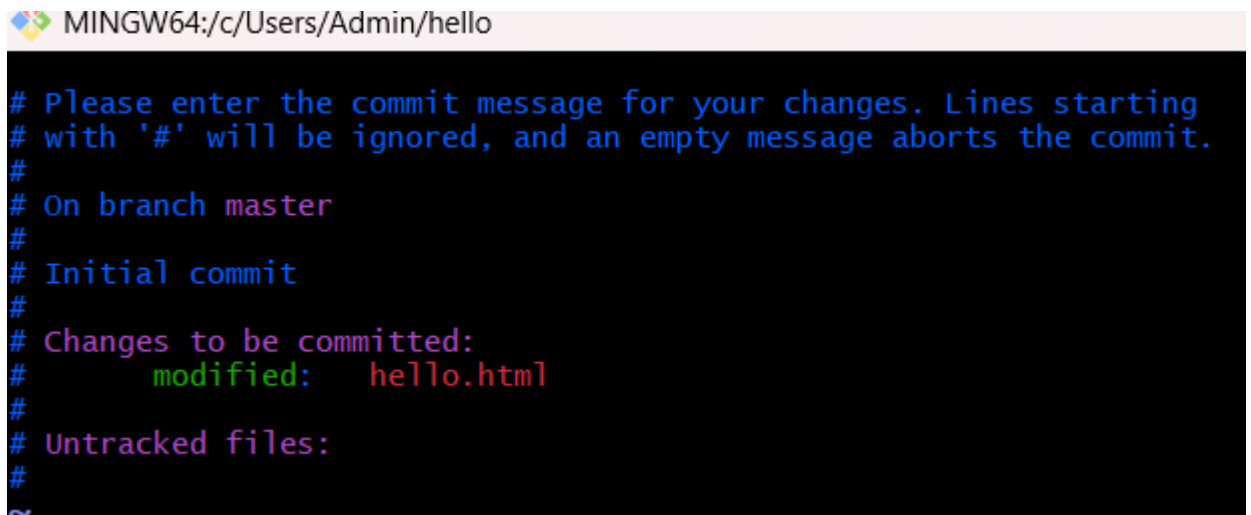
Для того, чтобы редактировать комментарий коммита, нужно использовать команду `git commit` без метки `-m`.



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

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
```

Рисунок 15 - Коммит изменений

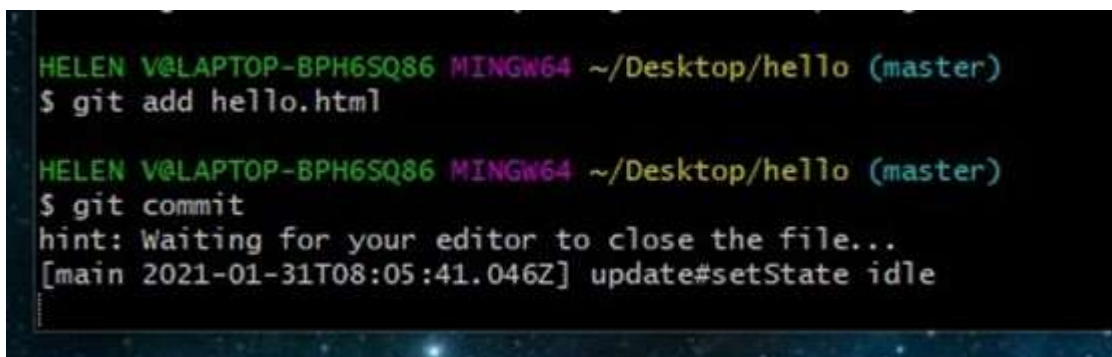


```
MINGW64:/c/Users/Admin/hello

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

Рисунок 16 - Ввод комментария

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



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

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
```

Рисунок 17 - Коммит-сообщения

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git status
On branch master
nothing to commit, working tree clean

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

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.

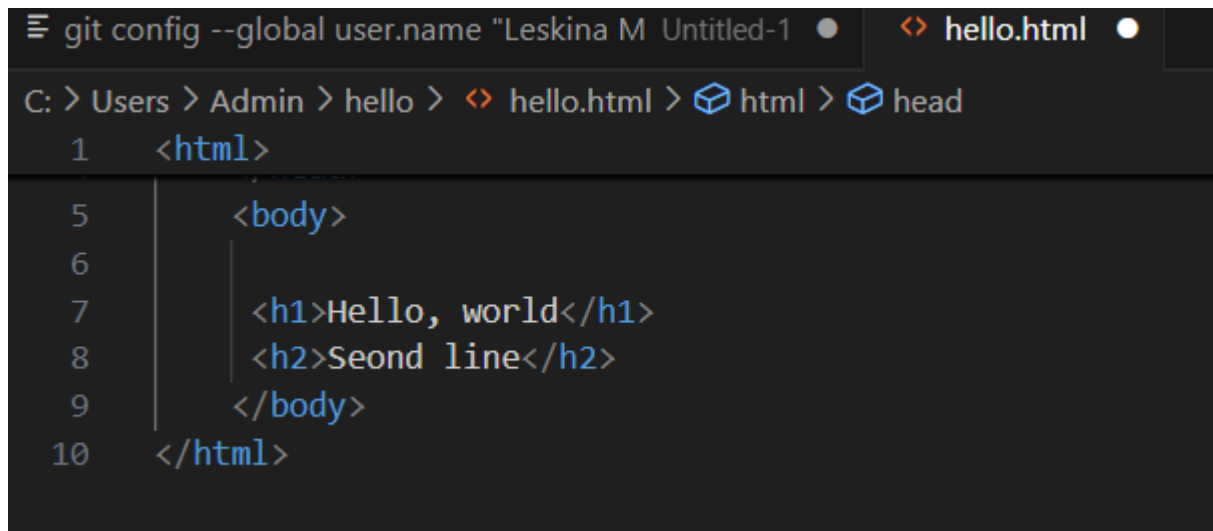
Сначала нужно в файл hello.html добавить теги html и body (рисунок 19), а затем проиндексировать изменения.



```
git config --global user.name "Leskina M"
C: > Users > Admin > hello > hello.html > html
1  <html>
2    <body>
3
4    <h1>Hello, world</h1>
5    <h2>Seond line</h2>
6    </body>
7  </html>
```

Рисунок 19 - Добавление тегов html и body

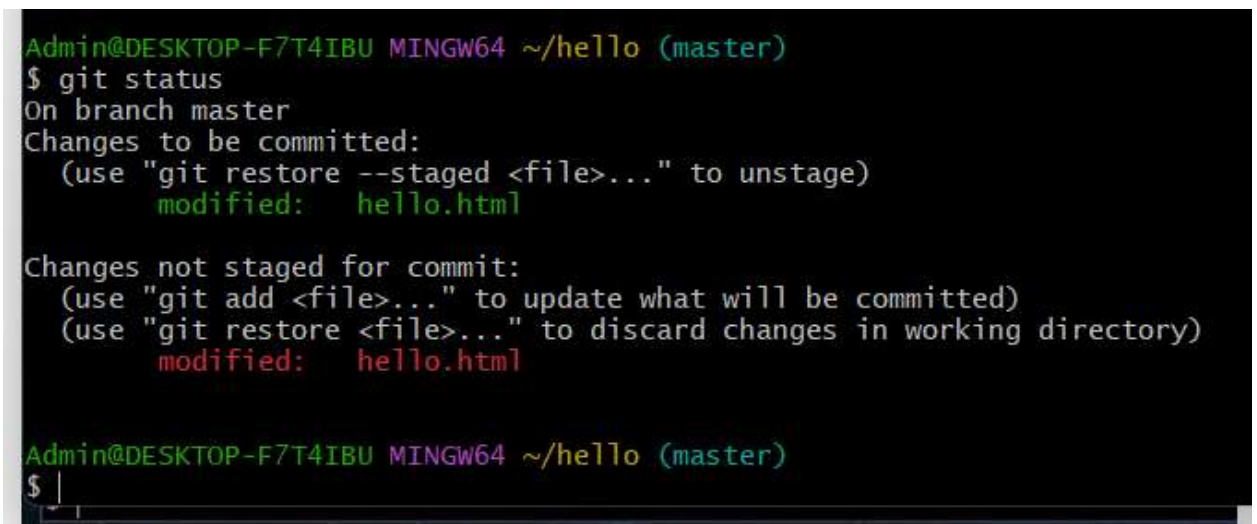
Затем еще раз нужно добавить изменения в файл (добавить тег head), но изменения не индексировать (рисунок 20).



```
git config --global user.name "Leskina M"
C: > Users > Admin > hello > hello.html > html > head
1 <html>
5     <body>
6     <h1>Hello, world</h1>
7     <h2>Seond 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.



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/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

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 21 - Проверка состояния

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


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Add standart HTML tags"
[master 3cfa397] Add standart HTML tags
1 file changed, 1 insertion(+)

Admin@DESKTOP-F7T4IBU MINGW64 ~/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")

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 22 - Коммит 1 изменения и проверка состояния

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



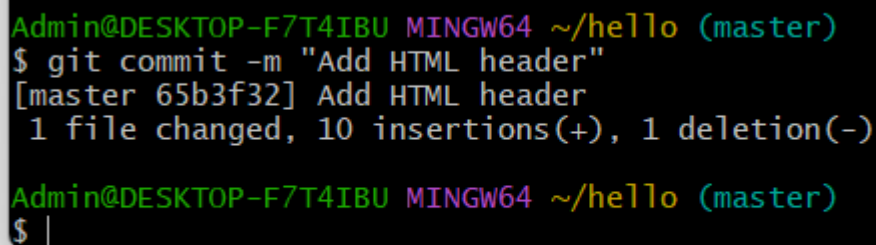
```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add .

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 23 - Индексация 2 изменения и проверка состояния

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



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Add HTML header"
[master 65b3f32] Add HTML header
1 file changed, 10 insertions(+), 1 deletion(-)

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 24 - Коммит 2 изменения

Гит 2

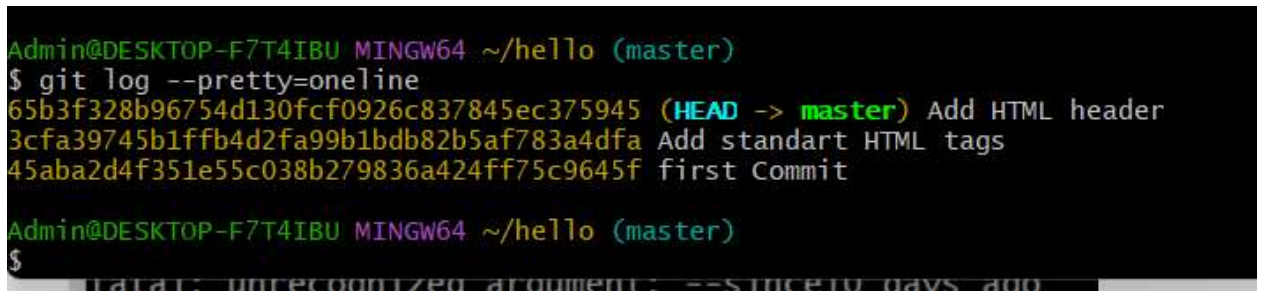
Работа с Git

История

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

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).

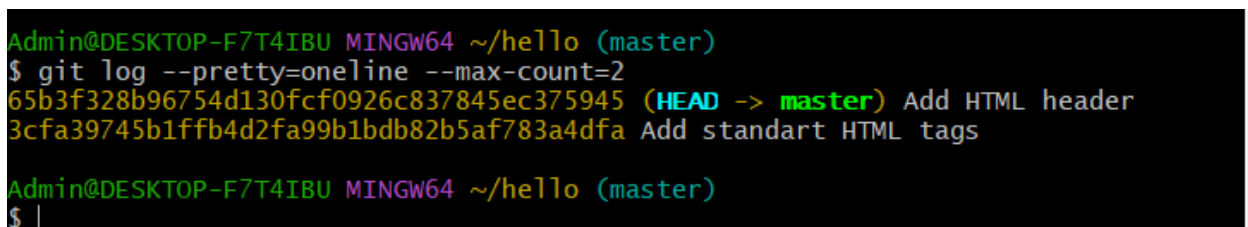


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git log --pretty=oneline
65b3f328b96754d130fcf0926c837845ec375945 (HEAD -> master) Add HTML header
3cfa39745b1ffb4d2fa99b1bdb82b5af783a4dfa Add standart HTML tags
45aba2d4f351e55c038b279836a424ff75c9645f first Commit

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 26 - Однострочный формат вывода

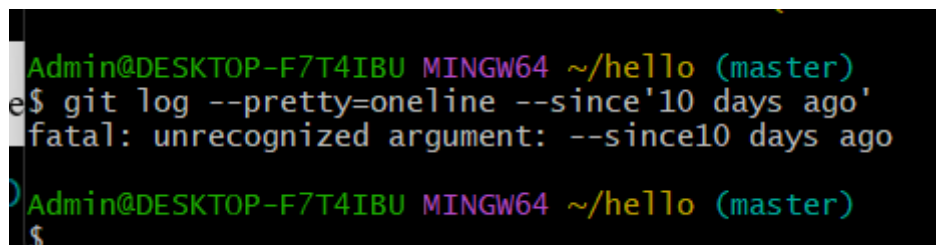
Далее на рисунках 3-8 показано несколько вариантов вывода истории изменений.



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git log --pretty=oneline --max-count=2
65b3f328b96754d130fcf0926c837845ec375945 (HEAD -> master) Add HTML header
3cfa39745b1ffb4d2fa99b1bdb82b5af783a4dfa Add standart HTML tags

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

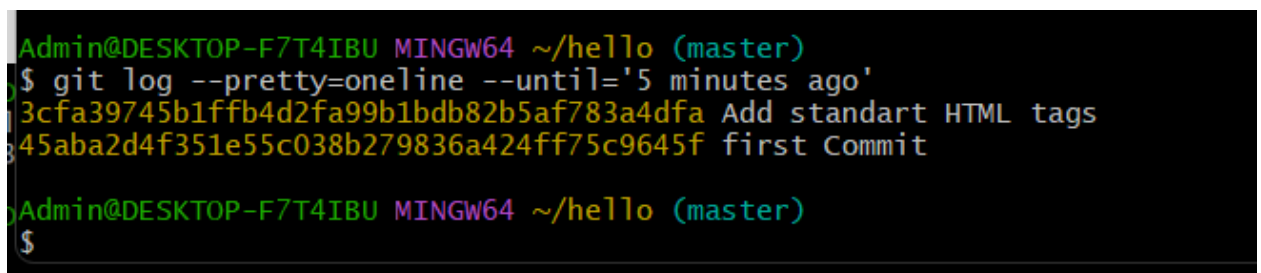
Рисунок 27 - Вывод последних 2 изменений



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git log --pretty=oneline --since=10 days ago
fatal: unrecognized argument: --since10 days ago

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

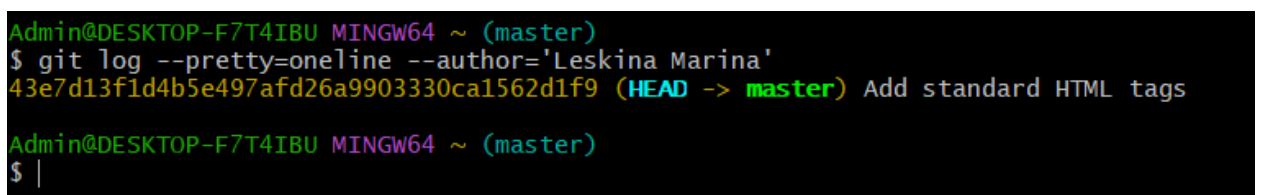
Рисунок 28 - Вывод изменений начиная с определенного времени



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git log --pretty=oneline --until='5 minutes ago'
3cfa39745b1ffb4d2fa99b1bdb82b5af783a4dfa Add standart HTML tags
45aba2d4f351e55c038b279836a424ff75c9645f first Commit

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 29 - Вывод изменений до определенного времени



```
Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ git log --pretty=oneline --author='Leskina Marina'
43e7d13f1d4b5e497afd26a9903330ca1562d1f9 (HEAD -> master) Add standard HTML tags

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$
```

Рисунок 30 - Вывод изменений, внесенных определенным автором

```
Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ git log --pretty=oneline --all
43e7d13f1d4b5e497afd26a9903330ca1562d1f9 (HEAD -> master) Add standard HTML tags
Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$
```

Рисунок 31 - Вывод всех изменений

```
Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ git log --pretty=format:"%h %cd %s (%an)" --since='12 days ago'
43e7d13 Tue Jun 3 16:52:19 2025 +0300 Add standard HTML tags (Leskina Marina)
Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$
```

Рисунок 32 - Использование нескольких параметров

Алиасы

Для настройки алиасов используется команда, показанная на рисунке 9.

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.co checkout
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.ci commit
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.st status
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.br blanch
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.hist "log --pretty=format:'%h %ad | %s%d [%an]' --graph --date=short"
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.type 'cat-file -t'
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.dump 'cat-file -p'
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 33 - Настройка алиасов для некоторых команд

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello1 (master)
$ git config --get alias.hist
log --oneline --graph --decorate --all

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello1 (master)
$ git hist

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello1 (master)
$
```

Рисунок 34 - Выполнение алиаса hist

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ alias gs='git status '

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ gs
On branch master
nothing to commit, working tree clean

Admin@DESKTOP-F7T4IBU MINGW64 ~/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).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git config --global alias.hist "log --pretty=format:'%h %ad | %s%d [%an]' --graph --date=short"

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist
* 65b3f32 2025-06-03 | Add HTML header (HEAD -> master) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git checkout 45aba2d
Note: switching to '45aba2d'.

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 45aba2d first Commit
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((45aba2d...))
$

```

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.

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((45aba2d...))
$ git checkout master
Previous HEAD position was 45aba2d first Commit
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ cat hello.html
<html>
  <head>

  </head>
  <body>

    <h1>Hello, world</h1>
    <h2>Seond line</h2>
  </body>
</html>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

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.

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag ver1

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 39 - Задание тега

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

```
$ git checkout ver1^
Note: switching to 'ver1^'.

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 3cfa397 Add standart HTML tags
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((3cfa397...))
$ |
```

Рисунок 40 – Переход к предыдущей версии с помощью тега

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((3cfa397...))
$ git tag ver1-beta

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$
```

Рисунок 41 - Задание тега предыдущей версии

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$ git checkout ver1
Previous HEAD position was 3cfa397 Add standart HTML tags
HEAD is now at 65b3f32 Add HTML header

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1))
$ git checkout ver1-beta
Previous HEAD position was 65b3f32 Add HTML header
HEAD is now at 3cfa397 Add standart HTML tags

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$

```

Рисунок 42 - Переключение между версиями с помощью тегов

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$ git tag
ver1
ver1-beta

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$

```

Рисунок 43 - Просмотр тегов

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$ git hist master --all
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1, master) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (HEAD, tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$

```

Рисунок 44 - Просмотр тегов в логе

Отмена локальных изменений (до индексации)

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

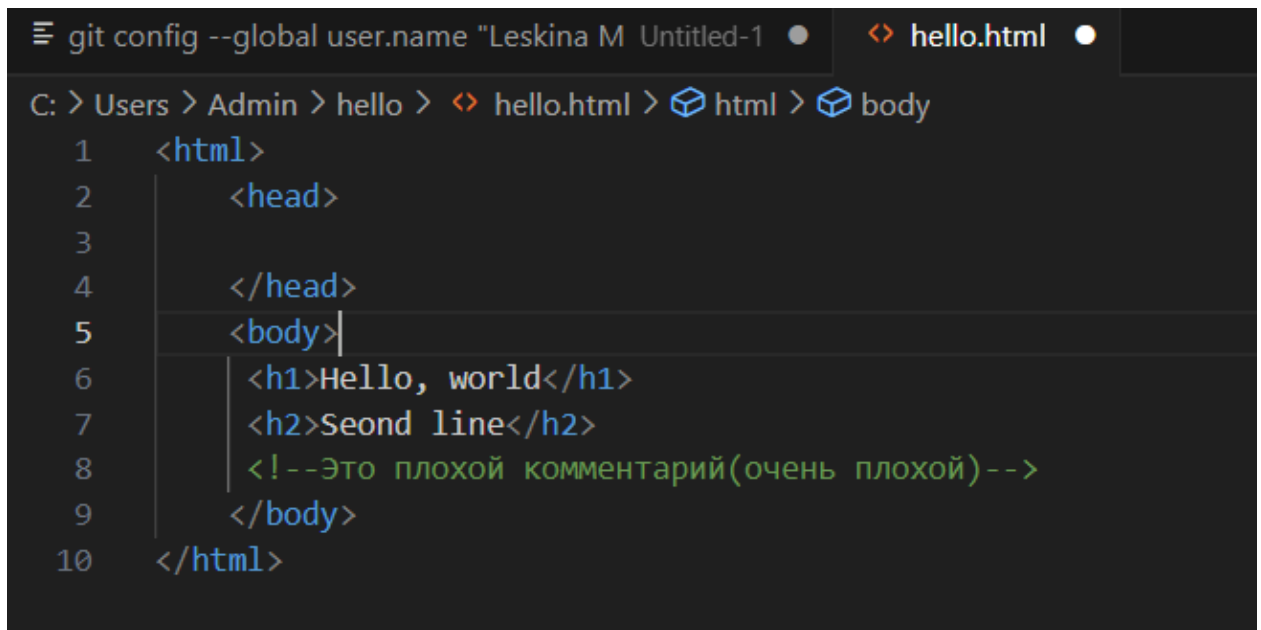
```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((ver1-beta))
$ git checkout master
Previous HEAD position was 3cfa397 Add standart HTML tags
Switched to branch 'master'

```

Рисунок 45 - Переключение на последний коммит

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



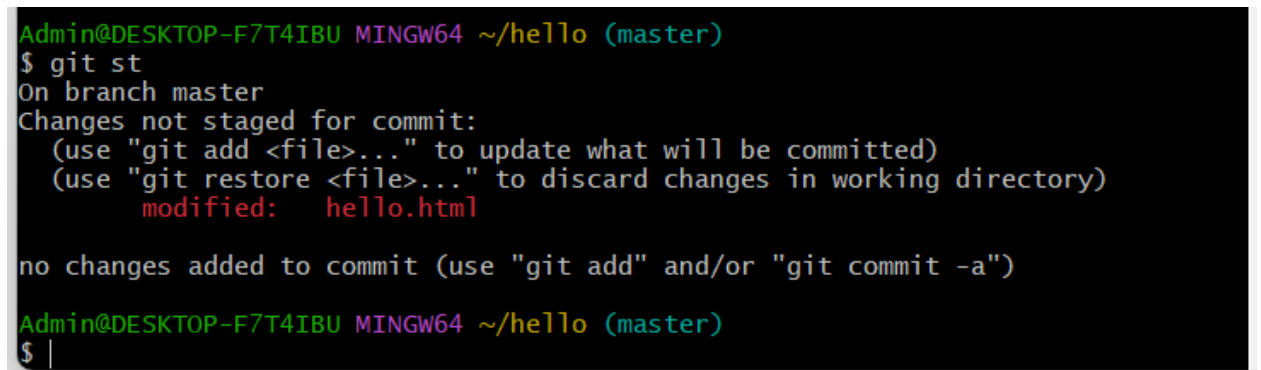
```
git config --global user.name "Leskina M"
Untitled-1
hello.html

C: > Users > Admin > hello > hello.html > html > body

1  <html>
2      <head>
3
4      </head>
5      <body>
6          <h1>Hello, world</h1>
7          <h2>Seond 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).



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/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")

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

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.

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git checkout hello.html
Updated 1 path from the index

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git st
On branch master
nothing to commit, working tree clean

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ cat hello.html
<html>
  <head>

  </head>
  <body>

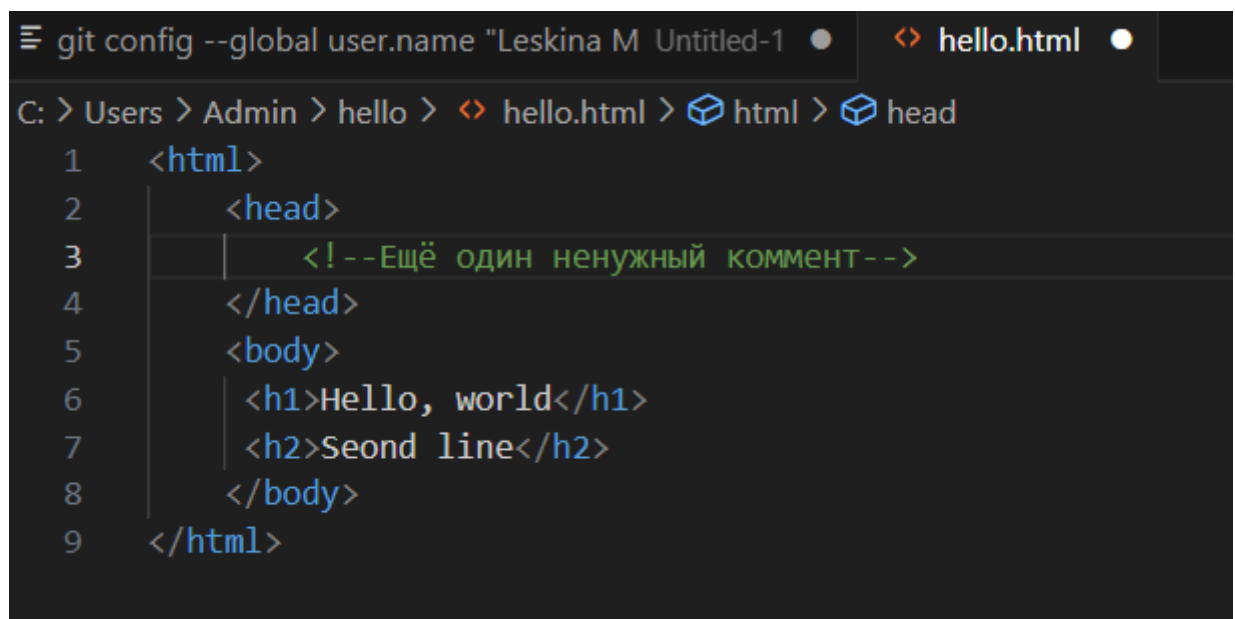
    <h1>Hello, world</h1>
    <h2>Seond line</h2>
  </body>
</html>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)

```

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).



```

git config --global user.name "Leskina M"
C: > Users > Admin > hello > hello.html > html > head
1  <html>
2    <head>
3      <!-- Ещё один ненужный коммент -->
4    </head>
5    <body>
6      <h1>Hello, world</h1>
7      <h2>Seond line</h2>
8    </body>
9  </html>

```

Рисунок 49 - Внесение ненужного изменения

```

</html>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git st
On branch master
nothing to commit, working tree clean

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/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")

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 51 - Очистка буферной зоны

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git checkout hello.html
Updated 1 path from the index

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git st
On branch master
nothing to commit, working tree clean

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 52 - Удаление ненужных изменений

Отмена коммитов

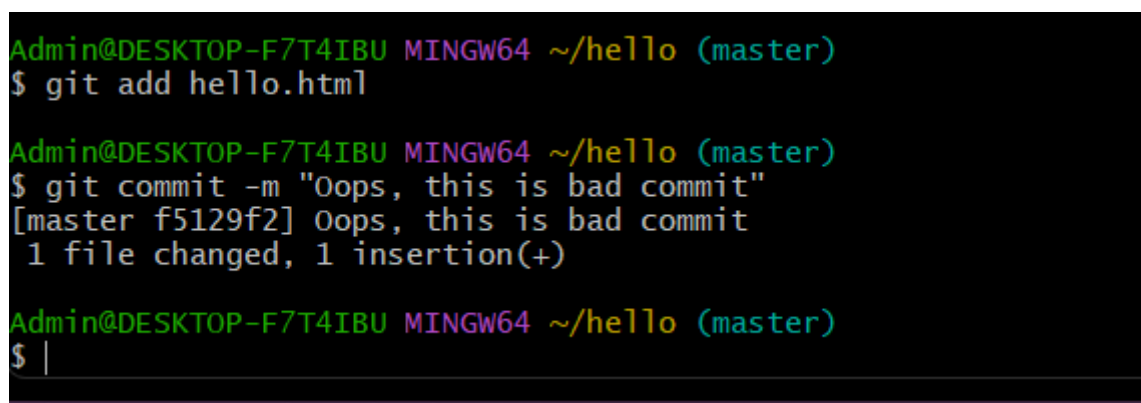
Для отмены коммита можно использовать способ создания нового коммита, отменяющего изменения.

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



```
git config --global user.name "Leskina M"
C:\Users\Admin\hello\hello.html
1 <html>
2   <head>
3
4   </head>
5   <body>
6
7   <h1>Hello, world</h1>
8   <h2>Seond line</h2>
9   <!--мяу-->
10  </body>
11 </html>
```

Рисунок 53 - Внесение изменения в файл



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Oops, this is bad commit"
[master f5129f2] Oops, this is bad commit
1 file changed, 1 insertion(+)

Admin@DESKTOP-F7T4IBU MINGW64 ~/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 "Reapply "Oops, this is bad commit""
This reverts commit a21409e83b00fd50a527ce8c94d7475a28cd029f.
# 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

```
[master a21409e] Reapply "Oops, this is bad commit"
1 file changed, 1 insertion(+)
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 57 - Редактор закрыт

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist
* 72841f9 2025-06-03 | Revert "Reapply "Oops, this is bad commit"" (HEAD -> mast
er) [Leskina Marina]
* a21409e 2025-06-03 | Reapply "Oops, this is bad commit" [Leskina Marina]
* b2fc2da 2025-06-03 | Revert "Oops, this is bad commit" [Leskina Marina]
* f5129f2 2025-06-03 | Oops, this is bad commit [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 58 - Все коммиты при просмотре лога

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

```

dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
git tag oooops

dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
git hist
72841f9 2025-06-03 | Revert "Reapply "Oops, this is bad commit"" (HEAD -> mas
r, tag: oooops) [Leskina Marina]
a21409e 2025-06-03 | Reapply "Oops, this is bad commit" [Leskina Marina]
b2fc2da 2025-06-03 | Revert "Oops, this is bad commit" [Leskina Marina]
f5129f2 2025-06-03 | Oops, this is bad commit [Leskina Marina]
65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
45aba2d 2025-06-03 | first Commit [Leskina Marina]

dmin@DESKTOP-F7T4IBU MINGW64 ~/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`.

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git reset --hard ver1
HEAD is now at 65b3f32 Add HTML header

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist
* 65b3f32 2025-06-03 | Add HTML header (HEAD -> master, tag: ver1) [Leskina Mari
na]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/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).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist --all
* 72841f9 2025-06-03 | Revert "Reapply "Oops, this is bad commit"" (tag: oooops)
  [Leskina Marina]
* a21409e 2025-06-03 | Reapply "Oops, this is bad commit" [Leskina Marina]
* b2fc2da 2025-06-03 | Revert "Oops, this is bad commit" [Leskina Marina]
* f5129f2 2025-06-03 | Oops, this is bad commit [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (HEAD -> master, tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/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).

```
dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
git tag -d oooops
deleted tag 'oooops' (was 72841f9)

dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
git hist --all
65b3f32 2025-06-03 | Add HTML header (HEAD -> master, tag: ver1) [Leskina Marina]
3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
45aba2d 2025-06-03 | first Commit [Leskina Marina]

dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
```

Рисунок 62 - Удаление тега


```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git type 4315008
commit

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git dump 4315008
tree 91ef51ec17b55120c4cb7905892047737c9c087c
parent ae8491dd99fc164e86f1b74d17be9a2510bce6bd
author Leskina Marina <linesssli9982gmail.com> 1749042858 +0300
committer Leskina Marina <linesssli9982gmail.com> 1749042858 +0300

Added index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |

```

Рисунок 63 - Использование алиасов

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git cat-file -p 91ef51e
100644 blob b203a5f120479c75cd8cdbfb20afe0991725647b    index.html
040000 tree c9cc182a87d56ff9bbc4bba9ecd152fe167e8dcf    lib

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |

```

Рисунок 64 - Просмотр дерева каталогов

Затем нужно посмотреть каталог lib (рисунок 21).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git cat-file -p c9cc182
100644 blob 8435c3df70fa4f8f3659c348329b3376e78ea996    hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |

```

Рисунок 65 - Просмотр каталога lib

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


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git cat-file -p 8435c3df
<!--крытааа-->
<html>
  <head>

  </head>
  <body>

    <h1>Hello, world</h1>
    <h2>The cake is a lie</h2>
  </body>
</html>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 66 - 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.

```
</html>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist
* 4315008 2025-06-04 | Added index.html (HEAD -> master) [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git cat-file -p 45aba2d
tree 0207049fcff52f60055434766127a023b1bc978e
author Leskina Marina <linesssli9982gmail.com> 1748969405 +0300
committer Leskina Marina <linesssli9982gmail.com> 1748969405 +0300

first Commit

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git cat-file -p 0207049
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    hello.html
```

Рисунок 67 - Просмотр содержимого файла при первом коммите

Работа с Git

Создание ветки

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git status
On branch style
Untracked files:
```

Рисунок 68 - Создание новой ветки style

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ touch lib/style.css

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$
```

Рисунок 69 - Создание файла стилей

```
al user.name "Leskina M" Untitled-1  hello.html C:\...\hello  h
C: > Users > Admin > hello > lib > # style.css > ...
1  h1 {
2  |   color: red;
3  }
```

Рисунок 70 - Код style.css

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git add lib/style.css

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git commit -m "Added css stylesheet"
[style 983d3ec] Added css stylesheet
1 file changed, 3 insertions(+)
create mode 100644 lib/style.css

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$
```

Рисунок 71 - Индексация и коммит нового файла

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

```
al user.name "Leskina M" Untitled-1  hello.html C:\...\hello  hello.html C:\...\lib  index.html  #
C: > Users > Admin > hello > lib > hello.html > html > head > link
1  <!--крутааа-->
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
```

Рисунок 72 - Изменения в файле hello.html

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git add lib/hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git commit -m "Hello.html uses style.css"
[style e8fb4de] Hello.html uses style.css
1 file changed, 1 insertion(+), 1 deletion(-)

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$

```

Рисунок 73 - Индексация и коммит

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

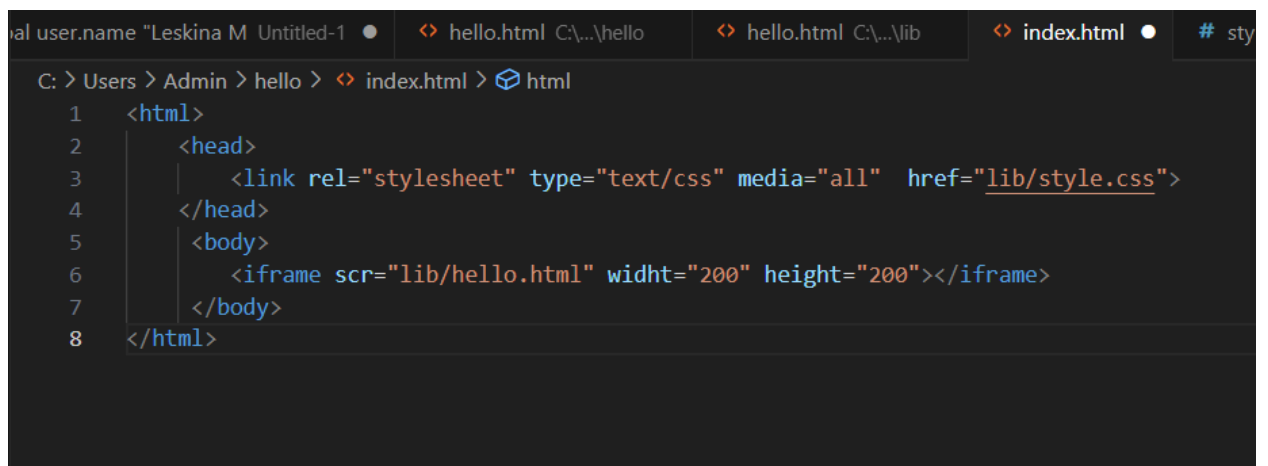


Рисунок 74 - Изменения в файле index.html

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git add index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git commit -m "Update index.html"
[style e236dce] Update index.html
1 file changed, 3 insertions(+)

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$

```

Рисунок 75 - 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.

```

dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
git hist --all
e236dce 2025-06-04 | Update index.html (HEAD -> style) [Leskina Marina]
e8fb4de 2025-06-04 | Hello.html uses style.css [Leskina Marina]
983d3ec 2025-06-04 | Added css stylesheet [Leskina Marina]
4315008 2025-06-04 | Added index.html (master) [Leskina Marina]
ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
45aba2d 2025-06-03 | first Commit [Leskina Marina]

dmin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)

```

Рисунок 76 - 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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ cat lib/hello.html
<!--крутааа-->
<html>
  <head>

  </head>
  <body>

    <h1>Hello, world</h1>
    <h2>The cake is a lie</h2>
  </body>
</html>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |

```

Рисунок 77 - Переключение на ветку master

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ cat lib/hello.html
<!--крутааа-->
<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>
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ |

```

Рисунок 78 - Переключение на ветку style

Изменения в ветке master

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

```

Рисунок 79 - Переключение на ветку master

```

git config --global user.name "Leskina M"
C: > Users > Admin > hello > README
1 This is the Hello World example the Git tutorial

```

Рисунок 80 - Содержимое файла README

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add README

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Added README"
[master afa49d9] Added README
1 file changed, 1 insertion(+)
create mode 100644 README

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |

```

Рисунок 81 - Индексация и коммит

Просмотр отличающихся веток

На рисунке 15 можно увидеть дерево коммитов.

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist --all
* afa49d9 2025-06-04 | Added README (HEAD -> master) [Leskina Marina]
| * e236dce 2025-06-04 | Update index.html (style) [Leskina Marina]
| * e8fb4de 2025-06-04 | Hello.html uses style.css [Leskina Marina]
| * 983d3ec 2025-06-04 | Added css stylesheet [Leskina Marina]
|/
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 82 - 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).

```
elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (master)
$ git checkout style
Switched to branch 'style'

elena@LAPTOP-I3I3QM5I MINGW64 ~/Documents/hello (style)
$ git merge master
```

Рисунок 83 - Переключение на ветку style

```
Merge made by the 'ort' strategy.
 README | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 README

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$
```

Рисунок 84 - Слияние с веткой master

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git hist --all
*   0df4740 2025-06-04 | Merge branch 'master' into style (HEAD -> style) [Leskina Marina]
| \
| * afa49d9 2025-06-04 | Added README (master) [Leskina Marina]
| * | e236dce 2025-06-04 | Update index.html [Leskina Marina]
| * | e8fb4de 2025-06-04 | Hello.html uses style.css [Leskina Marina]
| * | 983d3ec 2025-06-04 | Added css stylesheet [Leskina Marina]
| /
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ |

```

Рисунок 85 - Просмотр истории

Создание конфликта

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

```

1 Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

```

Рисунок 86 - Переход в ветку master

```

C: > Users > Admin > hello > lib > < > hello.html > < > html > < > head
1  <!--крутааа-->
2  <html>
3      <head>
4      | <!--мяуее-->
5      </head>
6      <body>
7      |
8      | <h1>Hello, world. Life is great</h1>
9      | <h2>The cake is a lie</h2>
10     </body>
11 </html>

```

Рисунок 87 - Внесение изменений


```
C: > Users > Admin > hello > lib > hello.html > html > body
1  <!--крутааа-->
2  <html>
3  |   <head>
4  |   |<=====> HEAD (Current Change)
5  |   |<link rel="stylesheet" type="text/css" media="all" href="style.css">
6  |   |=====
7  |   |<!--мняе-->
8  |>>>>>> master (Incoming Change)
9  |   </head>
10 |   <body>
11 |   |
12 |   |<h1>Hello, world. Life is great</h1>
13 |   |<h2>The cake is a lie</h2>
14 |   </body>
15 </html>
```

Рисунок 91 - Просмотр файла hello.html при наличии конфликта

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

```
C: > Users > Admin > hello > lib > hello.html > html > head > link
1  <!--крутааа-->
2  <html>
3  |   <head>
4  |   |<link rel="stylesheet" type="text/css" media="all" href="style.css">|
5  |   </head>
6  |   <body>
7  |   |<h1>Hello, world. Life is great</h1>
8  |   |<h2>The cake is a lie</h2>
9  |   </body>
10 </html>
```

Рисунок 92 - Решение конфликта вручную

Затем следует произвести индексацию и коммит (рисунок 26).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style|MERGING)
$ git add lib/hello.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style|MERGING)
$ git commit -m "Merges master fixed conflict."
[style 6e0ea73] Merges master fixed conflict.
```

Рисунок 93 - Индексация и коммит

Работа с Git

Сброс ветки style

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git checkout style
Already on 'style'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git hist
* 6e0ea73 2025-06-04 | Merges master fixed conflict. (HEAD -> style) [Leskina Marina]
|
| * 76bfe62 2025-06-04 | Life is great! (master) [Leskina Marina]
| * | 0df4740 2025-06-04 | Merge branch 'master' into style [Leskina Marina]
| \
| * afa49d9 2025-06-04 | Added README [Leskina Marina]
| * | e236dce 2025-06-04 | Update index.html [Leskina Marina]
| * | e8fb4de 2025-06-04 | Hello.html uses style.css [Leskina Marina]
| * | 983d3ec 2025-06-04 | Added css stylesheet [Leskina Marina]
| /
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]
```

Рисунок 94 - Просмотр истории

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git reset --hard 4315008
HEAD is now at 4315008 Added index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git hist --all
* 76bfe62 2025-06-04 | Life is great! (master) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html (HEAD -> style) [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
```

Рисунок 95 - Сброс ветки style

Сброс ветки master

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist
* 76bfe62 2025-06-04 | Life is great! (HEAD -> master) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html (style) [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 96 - Переключение на master и просмотр истории

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git reset --hard 76bfe62
HEAD is now at 76bfe62 Life is great!

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist --all
* 76bfe62 2025-06-04 | Life is great! (HEAD -> master) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html (style) [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 97 - Сброс ветки master

Перебазирование

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git checkout style
Switched to branch 'style'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git rebase master
Successfully rebased and updated refs/heads/style.

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git hist
* 76bfe62 2025-06-04 | Life is great! (HEAD -> style, master) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ |

```

Рисунок 98 - Перебазирование веток

Слияние в ветку master

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git merge style
Already up to date.

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |

```

Рисунок 99 - Слияние веток

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git hist
* 76bfe62 2025-06-04 | Life is great! (HEAD -> master, style) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 100 - 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.

```

$ cd

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ pwd
/c/Users/Admin

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ cd Documents
bash: cd: Documents: No such file or directory

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ git clone hello cloned_hello
Cloning into 'cloned_hello'...
done.

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ ls
-1.14-windows.xml
AppData/
'Application Data'@
Contacts/
Cookies@
Downloads/

```

Рисунок 101 - Переход в рабочий каталог и его клонирование

Просмотр клонированного репозитория

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ cd cloned_hello

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ ls
README  index.html  lib/

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$
```

Рисунок 102 - Просмотр содержимого клонированного репозитория

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git hist --all
* 76bfe62 2025-06-04 | Life is great! (HEAD -> master, origin/style, origin/master, origin/HEAD) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ |
```

Рисунок 103 - Просмотр логов клонированного каталога

Origin

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


```

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git remote
origin

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git remote show origin
* remote origin
  Fetch URL: C:/Users/Admin/hello
  Push URL: C:/Users/Admin/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)

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ |

```

Рисунок 104 - Просмотр данных об origin

Удаленные ветки

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

```

dmin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
git branch
master

dmin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
git branch -a
master
remotes/origin/HEAD -> origin/master
remotes/origin/master
remotes/origin/style

dmin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)

```

Рисунок 105 - Просмотр удаленных веток

Работа с Git

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


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ cd ../hello

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 106 - Переход в оригинальный репозиторий

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

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

Рисунок 107 - Изменения в файле README

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add README

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Changed README in original repo"
[master 13e6f84] Changed README in original repo
1 file changed, 2 insertions(+), 1 deletion(-)

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 108 - 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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ cd ../cloned_hello

Admin@DESKTOP-F7T4IBU MINGW64 ~/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 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 373 bytes | 31.00 KiB/s, done.
From C:/Users/Admin/hello
   76bfe62..13e6f84  master    -> origin/master

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$

```

Рисунок 109 - Извлечение изменений

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git hist --all
* 13e6f84 2025-06-05 | Changed README in original repo (origin/master, origin/HEAD) [Leskina Marina]
* 76bfe62 2025-06-04 | Life is great! (HEAD -> master, origin/style) [Leskina Marina]
* afa49d9 2025-06-04 | Added README [Leskina Marina]
* 4315008 2025-06-04 | Added index.html [Leskina Marina]
* ae8491d 2025-06-03 | Moved hello.html to lib [Leskina Marina]
* a951489 2025-06-03 | Add an author/email comment [Leskina Marina]
* 65b3f32 2025-06-03 | Add HTML header (tag: ver1) [Leskina Marina]
* 3cfa397 2025-06-03 | Add standart HTML tags (tag: ver1-beta) [Leskina Marina]
* 45aba2d 2025-06-03 | first Commit [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$

```

Рисунок 110 - Просмотр истории

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ cat README
This is the Hello world example the Git tutorial
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ |

```

Рисунок 111 - Вывод содержимого файла README

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

```

This is the Hello World example the Git tutorial
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git merge origin/master
Updating 76bfe62..13e6f84
Fast-forward
 README | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ |

```

Рисунок 112 - Слияние изменений

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ cat README
This is the Hello World example the Git tutorial
(changed in original)
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ |

```

Рисунок 113 - Вывод содержимого файла README

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

```

pull
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git pull
Already up to date.

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ |

```

Рисунок 114 - Команда git pull

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git branch --track style origin/style
branch 'style' set up to track 'origin/style'.

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git branch -a
master
style
remotes/origin/HEAD -> origin/master
remotes/origin/master
remotes/origin/style

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git hist --max-count=2
13e6f84 2025-06-05 | Changed README in original repo (HEAD -> master, origin/m
ster, origin/HEAD) [Leskina Marina]
76bfe62 2025-06-04 | Life is great! (origin/style, style) [Leskina Marina]

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$

```

Рисунок 115 - Добавление локальной ветки

Далее необходимо создать чистый репозиторий (рисунок 11).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ cd ..

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ git clone --bare hello hello.git
Cloning into bare repository 'hello.git'...
done.

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ ls hello.git
HEAD config description hooks/ info/ objects/ packed-refs refs/

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$

```

Рисунок 116 - Создание чистого репозитория

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ cd hello

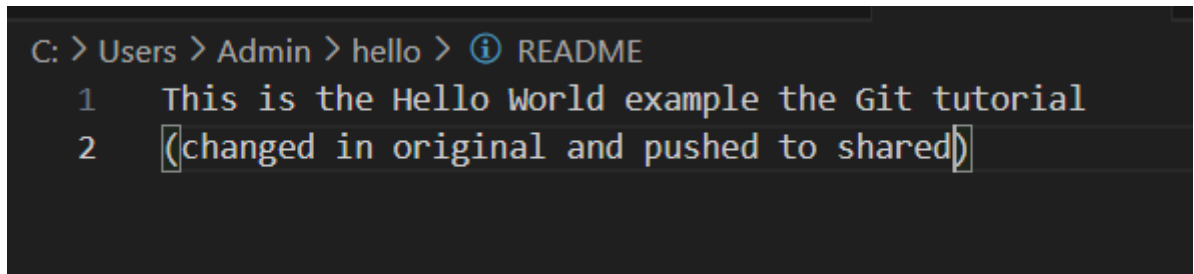
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git remote add shared ../hello.git

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 117 - Добавление удаленного репозитория

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



```
C: > Users > Admin > hello > ⓘ README
1 This is the Hello World example the Git tutorial
2 [(changed in original and pushed to shared)]
```

Рисунок 118 - Внесение изменений в файл



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git checkout master
M README
Already on 'master'

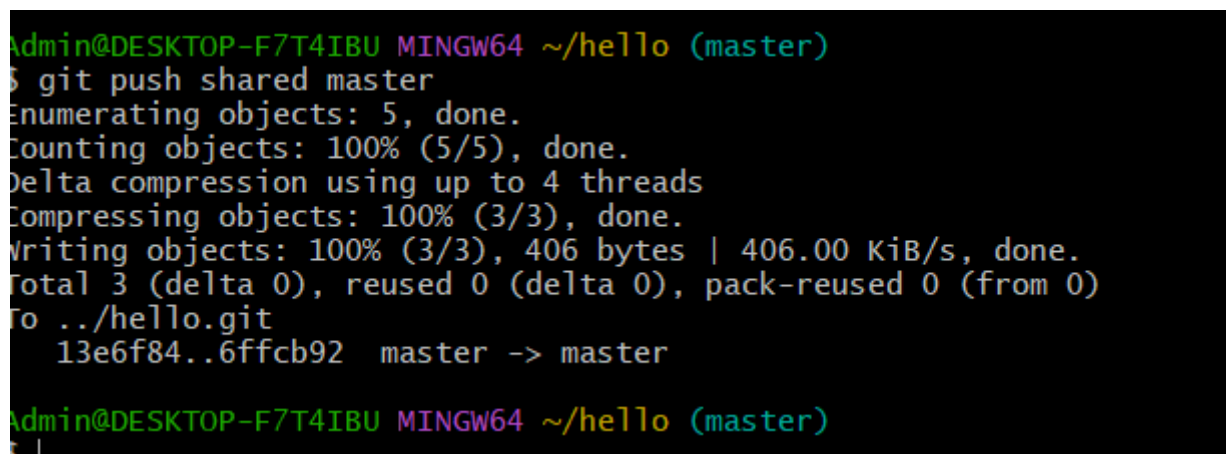
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git add README

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git commit -m "Added shared comment to README"
[master 6ffcb92] Added shared comment to README
1 file changed, 1 insertion(+), 1 deletion(-)

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 119 - Индексация и коммит

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



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/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), 406 bytes | 406.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To ../hello.git
13e6f84..6ffcb92 master -> master

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 120 - Отправка изменений в общий репозиторий

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ cd ../cloned_hello

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$
```

Рисунок 121 - Переход в клонированный репозиторий

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git remote add shared ../hello.git

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git branch --track shared master
branch 'shared' set up to track 'master'.

Admin@DESKTOP-F7T4IBU MINGW64 ~/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 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 386 bytes | 32.00 KiB/s, done.
From ../hello
 * branch                master      -> FETCH_HEAD
 * [new branch]           master      -> shared/master
Updating 13e6f84..6ffcb92
Fast-forward
 README | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ cat README
This is the Hello World example the Git tutorial
(changed in original and pushed to shared)
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$
```

Рисунок 122 - 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).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/cloned_hello (master)
$ git daemon --verbose --export-all --base-path=.
[5224] Ready to rumble
```

Рисунок 123 - Настройка сервера

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git clone ~/hello.git network_hello
Cloning into 'network_hello'...
done.

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ cd network_hello

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/network_hello (master)
$ ls
README  index.html  lib/

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/network_hello (master)
$

```

Рисунок 124 - Клонирование проекта

Работа с Git

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ 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 (from 1)
Receiving objects: 100% (1857/1857), 334.06 KiB | 1.92 MiB/s, done.
Resolving deltas: 100% (837/837), done.

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ cd ticgit

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote
origin

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ |

```

Рисунок 125 – Клонирование репозитория и просмотр удаленных репозиторий

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


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote -v
origin https://github.com/schacon/ticgit (fetch)
origin https://github.com/schacon/ticgit (push)

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ |
```

Рисунок 126 - Просмотр удаленных репозитория с ключом -v

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote add pb https://github.com/paulboone/ticgit

Admin@DESKTOP-F7T4IBU MINGW64 ~/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)

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$
```

Рисунок 127 - Добавление удаленного репозитория

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git fetch pb
remote: Enumerating objects: 43, done.
remote: Counting objects: 100% (22/22), done.
remote: Total 43 (delta 22), reused 22 (delta 22), pack-reused 21 (from 1)
Unpacking objects: 100% (43/43), 5.99 KiB | 39.00 KiB/s, done.
From https://github.com/paulboone/ticgit
* [new branch]      master    -> pb/master
* [new branch]      ticgit    -> pb/ticgit

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ |
```

Рисунок 128 - Использование имени вместо пути

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


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git fetch ~/cloned_hello
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 30 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (30/30), 2.79 KiB | 29.00 KiB/s, done.
From C:/Users/Admin/cloned_hello
* branch          HEAD          -> FETCH_HEAD
```

Рисунок 129 - Получение данных из удаленных проектов

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

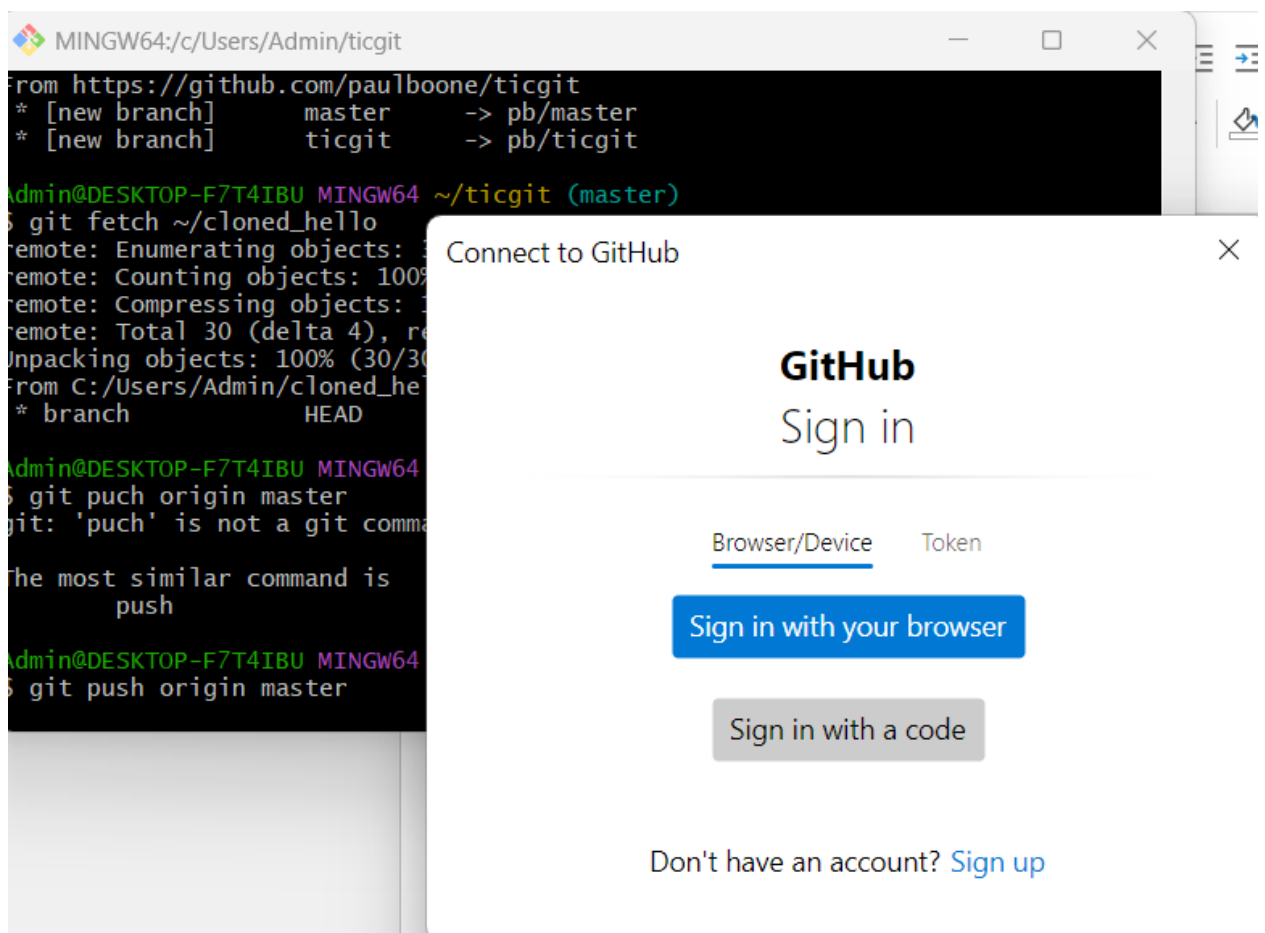


Рисунок 130 - Отправка изменений в удаленный репозиторий

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~ (master)
$ cd ticgit

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote show origin
* remote origin
Fetch URL: https://github.com/schacon/ticgit
Push URL: https://github.com/schacon/ticgit
HEAD branch: master
Remote branches:
  master tracked
  ticgit 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)

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ |

```

Рисунок 131 - Информация об удаленном репозитории

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote rename pd paul
error: No such remote: 'pd'

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote
origin

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ |

```

Рисунок 132 - Переименование удаленного репозитория

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

```

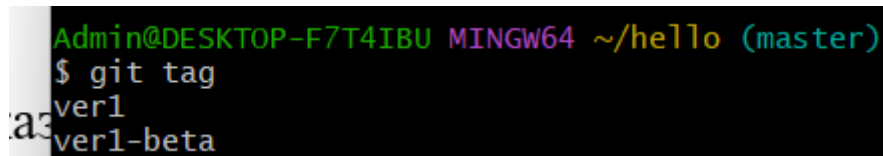
error: No such remote: 'pd'

Admin@DESKTOP-F7T4IBU MINGW64 ~/ticgit (master)
$ git remote
origin

```

Рисунок 133 - Удаление удаленного репозитория

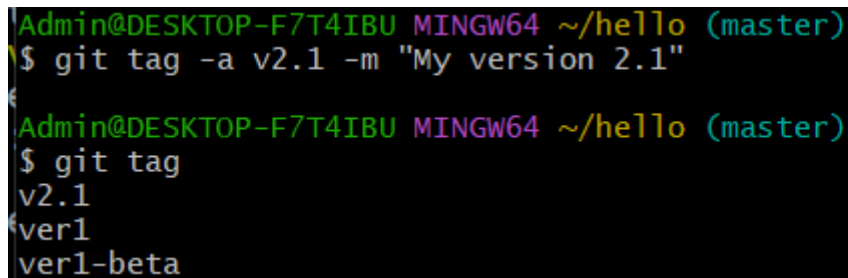
Просмотреть существующие теги можно с помощью команды `git tag` (рисунок 10).



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag
ver1
ver1-beta
```

Рисунок 134 - Просмотр тегов

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

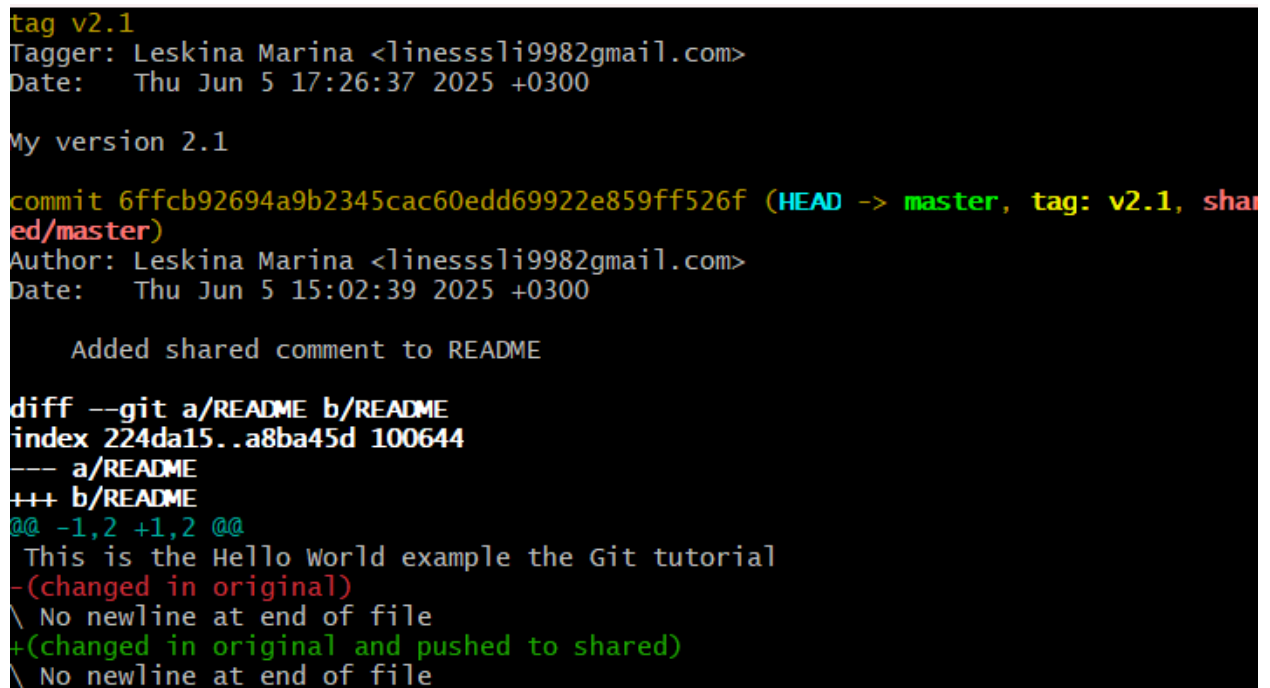


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag -a v2.1 -m "My version 2.1"

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag
v2.1
ver1
ver1-beta
```

Рисунок 135 - Создание аннотированной метки

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



```
tag v2.1
Tagger: Leskina Marina <linesssli9982gmail.com>
Date: Thu Jun 5 17:26:37 2025 +0300

My version 2.1

commit 6ffcb92694a9b2345cac60edd69922e859ff526f (HEAD -> master, tag: v2.1, shared/master)
Author: Leskina Marina <linesssli9982gmail.com>
Date: Thu Jun 5 15:02:39 2025 +0300

    Added shared comment to README

diff --git a/README b/README
index 224da15..a8ba45d 100644
--- a/README
+++ b/README
@@ -1,2 +1,2 @@
 This is the Hello World example the Git tutorial
-(changed in original)
 \ No newline at end of file
+(changed in original and pushed to shared)
 \ No newline at end of file
```

Рисунок 136 - 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).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag v2.1-1w

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag
v2.1
v2.1-1w
ver1
ver1-beta
```

Рисунок 137 - Создание легковесной метки

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git show v2.1-1w
commit 6ffcb92694a9b2345cac60edd69922e859ff526f (HEAD -> master, tag: v2.1-1w, tag: v2.1, shared/master)
Author: Leskina Marina <linesssli9982gmail.com>
Date: Thu Jun 5 15:02:39 2025 +0300

    Added shared comment to README

diff --git a/README b/README
index 224da15..a8ba45d 100644
--- a/README
+++ b/README
@@ -1,2 +1,2 @@
 This is the Hello World example the Git tutorial
-(changed in original)
 \ No newline at end of file
+(changed in original and pushed to shared)
 \ No newline at end of file

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$
```

Рисунок 138 - Просмотр данных тега

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git log --pretty=oneline
6ffcb92694a9b2345cac60edd69922e859ff526f (HEAD -> master, tag: v2.1-1w, tag: v2.1, shared/master) Added shared comment to README
13e6f8496ef492e0483e0ed9e329b63702c84307 Changed README in original repo
76bfe621f031402b182d55a44b1e23c95d961986 (style) Life is great!
afa49d9838c60413327d0391a21b2ad9f66cc213 Added README
4315008ba3474e087c0b71f6445e24a1e1d44270 Added index.html
ae8491dd99fc164e86f1b74d17be9a2510bce6bd Moved hello.html to lib
a951489e454e1cc9dfcf54a230ac9bab577d69b2 Add an author/email comment
65b3f328b96754d130fcf0926c837845ec375945 (tag: ver1) Add HTML header
3cfa39745b1fffb4d2fa99b1bdb82b5af783a4dfa (tag: ver1-beta) Add standart HTML tags
45aba2d4f351e55c038b279836a424ff75c9645f first Commit

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 139 - Просмотр истории

```

elena@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 140 - 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 141 - Ввод сообщения в текстовом редакторе

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

```

Tag message1

commit 13e6f8496ef492e0483e0ed9e329b63702c84307 (tag: ver3)
Author: Leskina Marina <linesssli9982gmail.com>
Date: Thu Jun 5 14:51:45 2025 +0300

    Changed README in original repo

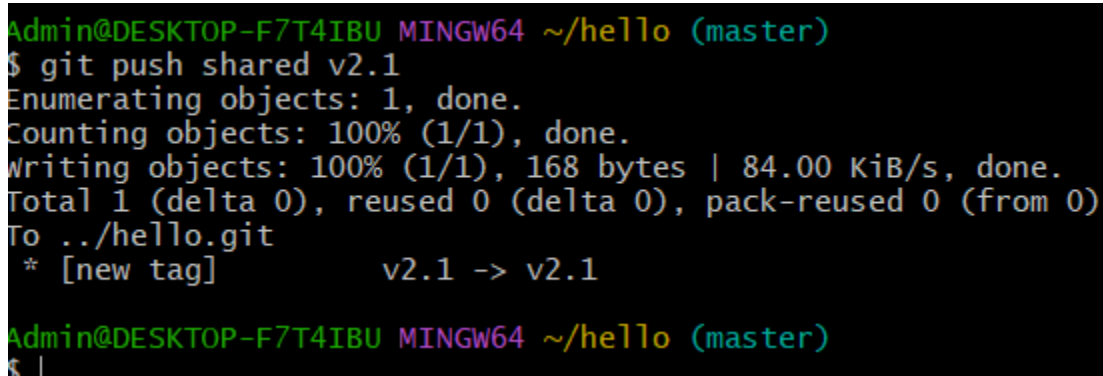
diff --git a/README b/README
index 9b6d5eb..224da15 100644
--- a/README
+++ b/README
@@ -1,2 @@
-This is the Hello World example the Git tutorial
\ No newline at end of file
+This is the Hello World example the Git tutorial
+(changed in original)
\ No newline at end of file

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)

```

Рисунок 142 - Просмотр данных тега

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

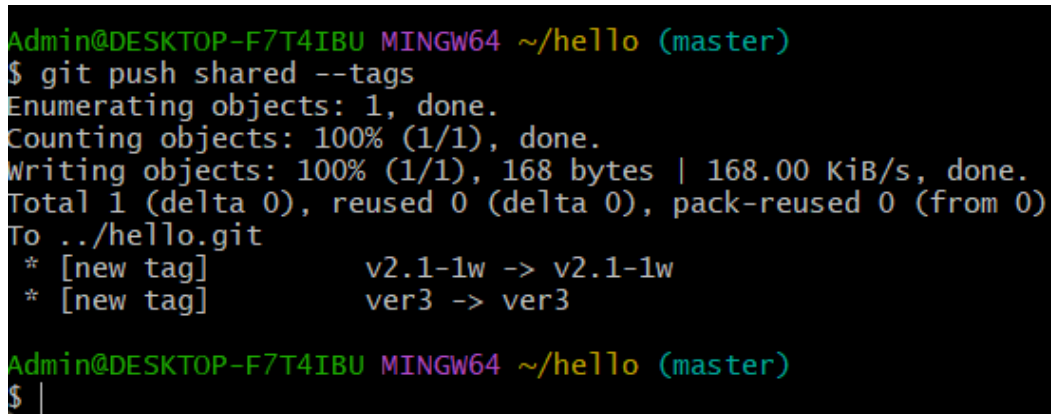


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git push shared v2.1
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 168 bytes | 84.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To ../hello.git
 * [new tag]          v2.1 -> v2.1

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 143 - Отправка тега на удаленный сервер

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

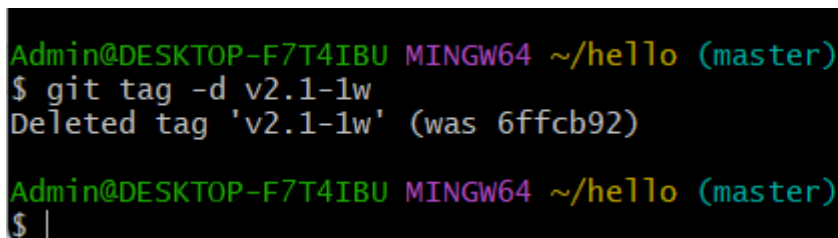


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git push shared --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 168 bytes | 168.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To ../hello.git
 * [new tag]          v2.1-1w -> v2.1-1w
 * [new tag]          ver3 -> ver3

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 144 - Отправка всех тегов на сервер

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



```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ git tag -d v2.1-1w
Deleted tag 'v2.1-1w' (was 6ffcb92)

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$ |
```

Рисунок 145 - 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 146 - 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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/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 6ffcb92 Added shared comment to README
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((v2.1))
$

```

Рисунок 147 - Переключение на метку

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello ((v2.1))
$ git checkout -b version2 v2.1
Switched to a new branch 'version2'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$

```

Рисунок 148 - Создание новой ветки

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.co checkout

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.br branch
```

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.ci commit

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.st status
```

Рисунок 149 - Задание алиасов

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.unstage 'reset HEAD --'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ |
```

Рисунок 150 - Создание псевдонима исключения файла из индекса

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git unstage fileA

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git reset HEAD -- fileA

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ |
```

Рисунок 151 - Использование созданного псевдонима

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.last 'log -1 HEAD'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ |
```

Рисунок 152 - Создание алиаса для просмотра последнего коммита


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git last
commit 6ffcb92694a9b2345cac60edd69922e859ff526f (HEAD -> version2, tag: v2.1, shared/master, master)
Author: Leskina Marina <linesssli9982gmail.com>
Date: Thu Jun 5 15:02:39 2025 +0300

    Added shared comment to README

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ |
```

Рисунок 153 - Результат работы созданного алиаса

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git config --global alias.visual '!gitk'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ |
```

Рисунок 154 - Создание псевдонима внешней команды

Работа с Git

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git add README test.rb LICENSE

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git commit -m "Initial commit of my project"
[master d875e84] Initial commit of my project
3 files changed, 2 insertions(+)
create mode 100644 hello.two/LICENSE
create mode 100644 hello.two/README
create mode 100644 hello.two/test.rb

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ |
```

Рисунок 1 - Индексация и коммит 3 файлов

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (version2)
$ git branch testing
```

Рисунок 2 - Создание ветки testing

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git checkout testing
D      LICENSE
D      README
D      test.rb
Switched to branch 'testing'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (testing)
$

```

Рисунок 3 - Переключение на ветку testing

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (testing)
$ git add test.rb

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (testing)
$ git commit -m 'New changes'
[testing 0434194] New changes
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 hello.two/test.rb

```

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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (testing)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (master)
$

```

Рисунок 5 - Переключение на ветку master

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (testing)
$ git checkout master
D      hello.two/test.rb
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$

```

Рисунок 6 - Еще индексация и коммит test.rb

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

```

create mode 100644 test.rb

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (mast
$ git checkout -b 'iss53'
Switched to a new branch 'iss53'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello (iss5
$

```

Рисунок 7 - Создание и переключение на ветку iss53

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (iss53)
a$ git add index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (iss53)
$ git commit -m 'Added html'
[iss53 b83b007] Added html
1 file changed, 5 insertions(+)
create mode 100644 hello.two/index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (iss53)
a$
commit -a -m 'Fixed the broken email address'
ix a935cd71 Fixed the broken email address

```

Рисунок 8 - Индексация и коммит файла index.html

Далее нужно переключить ветку на master (рисунок 9).

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).

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (hotfix)
$ git add index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (hotfix)
$ git commit -a -m 'Fixed the broken email address'
[hotfix df4eec0] Fixed the broken email address
1 file changed, 5 deletions(-)
delete mode 100644 hello.two/index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (hotfix)
$

```

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (hotfix)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git merge hotfix
Updating 84721e5..df4eec0
Fast-forward
 hello.two/index.html | 5 -----
 1 file changed, 5 deletions(-)
 delete mode 100644 hello.two/index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$

```

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

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch -d hotfix
Deleted branch hotfix (was df4eec0).

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$

```

Рисунок 12 - Удаление ветки hotfix

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

```

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (iss53)
$ git add index.html

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (iss53)
$ git commit -a -m 'Finished the new footer'
[iss53 2177f38] Finished the new footer
 1 file changed, 5 deletions(-)
 delete mode 100644 hello.two/index.html

```

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

```

Merge made by the 'ort' strategy.
 index.html | 5 -----
 1 file changed, 5 deletions(-)
 delete mode 100644 index.html

```

Рисунок 14 - Слияние веток

После этого ветку iss53 нужно удалить (рисунок 15).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch -d iss53
Deleted branch iss53 (was 2177f38).
```

Рисунок 15 - Удаление ветки iss53

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).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch
* master
  style
  testing
```

Рисунок 155 - Список существующих веток

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch -v
* master    4751f9a Merge branch 'iss53'
  style     76bfe62 Life is great!
  testing   0434194 New changes
```

Рисунок 156 - 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).

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch --merged
* master
  style
```

Рисунок 157 - Список веток слитых с текущей

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch --no-merged
```

Рисунок 158 - Список веток не слитых с текущей

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch -D testing
Deleted branch testing (was 0434194).
```

Рисунок 159 - Удаление ветки

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ git remote add origin https://github.com/Kanodov21/git-1.9.git
error: remote origin already exists.

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ git remote show origin
fatal: User cancelled dialog.
error: unable to read askpass response from 'C:/Program Files/Git/mingw64/bin/git-askpass.exe'
Username for 'https://github.com':
```

Рисунок 160 - Просмотр удаленных веток

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ git push origin main
info: please complete authentication in your browser...
remote: Repository not found.
fatal: repository 'https://github.com/Kanodov21/git-1.9.git/' not found

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ |
```

Рисунок 161 - Отправка изменений

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ git fetch origin
fatal: User cancelled dialog.
error: unable to read askpass response from 'C:/Program Files/Git/mingw64/bin/git-askpass.exe'
Username for 'https://github.com': |
```

Рисунок 162 - Выполнение команды git fetch

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ git checkout -b serv origin/main
fatal: 'origin/main' is not a commit and a branch 'serv' cannot be created from it
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
```

Рисунок 163 - Создание ветки на основе удаленной ветки

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

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (main)
$ git checkout serv
Switched to branch 'serv'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (serv)
$ git push origin :serv
```

Рисунок 164 - 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.


```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (master)
$ git checkout exp
Switched to branch 'exp'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two/repo1 (exp)
$ git rebase master
Current branch exp is up to date.
```

Рисунок 165 - 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.

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git rebase --onto master se ls
fatal: invalid upstream 'se'
```

Рисунок 166 - Перемещение изменений с параметром onto

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (cl)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git merge cl
Already up to date.
```

Рисунок 167 - Слияние веток master и cl

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git rebase master se
fatal: no such branch/commit 'se'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
```

Рисунок 168 - Перемещение изменений

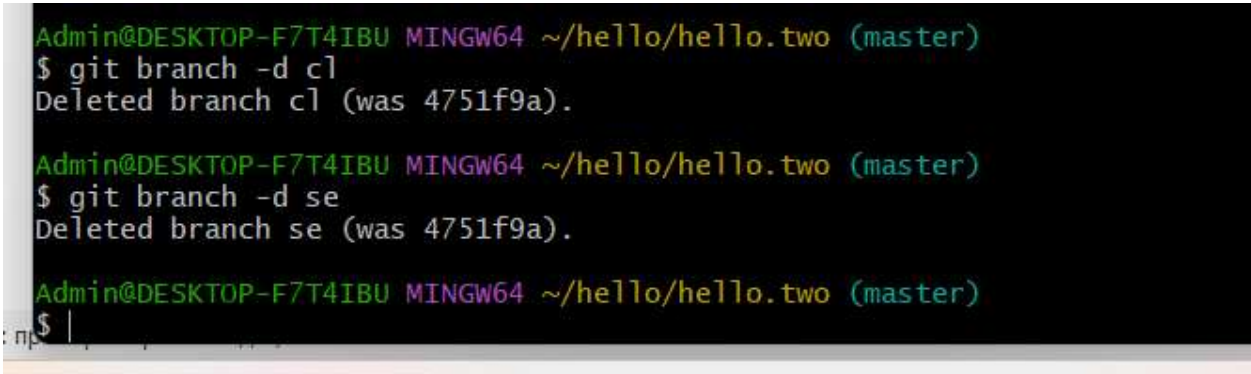
```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (se)
$ git checkout master
Switched to branch 'master'

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git merge se
Already up to date.

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ |
```

Рисунок 169 - Слияние веток master и se

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

A screenshot of a terminal window with a black background and green text. The prompt is 'Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)'. The first command is '\$ git branch -d cl', followed by the output 'Deleted branch cl (was 4751f9a)'. The second command is '\$ git branch -d se', followed by the output 'Deleted branch se (was 4751f9a)'. The third command is '\$', followed by a cursor. The terminal window has a light gray title bar at the top.

```
Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch -d cl
Deleted branch cl (was 4751f9a).

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ git branch -d se
Deleted branch se (was 4751f9a).

Admin@DESKTOP-F7T4IBU MINGW64 ~/hello/hello.two (master)
$ |
```

Рисунок 170 - Удаление веток cl и se