

Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и  
радиоэлектроники»

Кафедра инженерной психологии и эргономики

**Пользовательские интерфейсы информационных систем**

Отчет по практическим занятиям на тему  
«Образовательный курс GitHowTo»

Выполнил:  
студент гр.  
210901  
Лойко А.И

Проверил:  
Давыдович К. И.

Минск 2024

**Цель:** сформировать понимание и специфику работы с инструментом контроля версий «Git» и научиться пользоваться его основным функционалом. Курс располагается по ссылке <https://githowto.com/ru>.

## Прохождение курса

### Часть I: Основы Git

#### Задание 1. Приготовления

Если вы никогда ранее не использовали Git, для начала вам необходимо осуществить установку. Выполните следующие команды, чтобы Git узнал ваше имя и электронную почту. Эти данные используются для подписи изменений сделанных вами, что позволит отслеживать, кто и когда сделал изменения в файле.

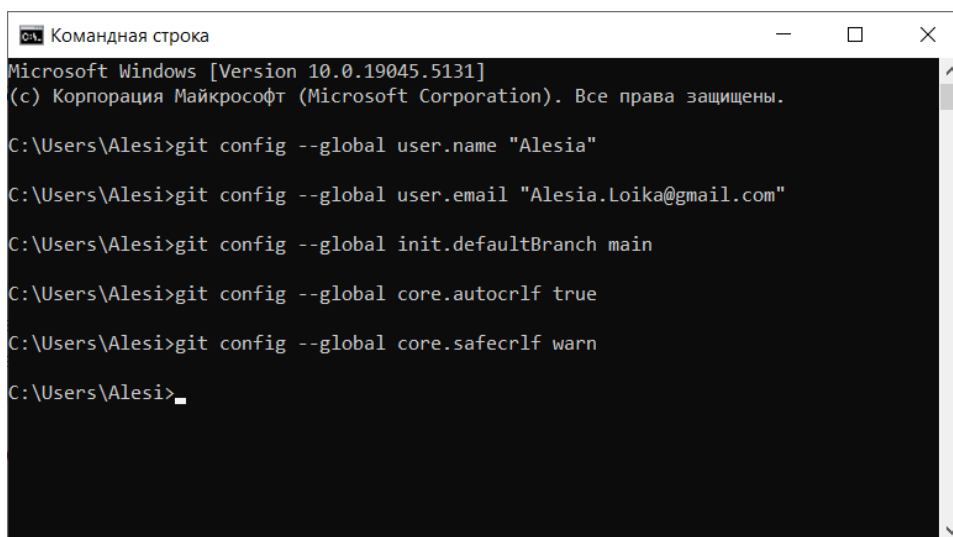
```
git config --global user.name "Your Name"
git config --global user.email "your_email@whatever.com"
```

Мы будем использовать main в качестве имени ветки по умолчанию. Чтобы установить его, выполните следующую команду:

```
git config --global init.defaultBranch main
```

Также нужно выполнить данные команды для корректной обработки окончаний строк (для Windows):

```
git config --global core.autocrlf true
git config --global core.safecrlf warn
```

A screenshot of a Windows Command Prompt window titled "Командная строка". The window shows the following commands and their execution: 

```
Microsoft Windows [Version 10.0.19045.5131]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Alesia>git config --global user.name "Alesia"

C:\Users\Alesia>git config --global user.email "Alesia.Loika@gmail.com"

C:\Users\Alesia>git config --global init.defaultBranch main

C:\Users\Alesia>git config --global core.autocrlf true

C:\Users\Alesia>git config --global core.safecrlf warn

C:\Users\Alesia>
```

Рисунок 1 – Начало работы с Git

## Задание 2. Создание проекта

Начните работу в пустой директории (например, repositories, если вы скачали архив с предыдущего шага) с создания пустой поддиректории work, затем войдите в неё и создайте там файл hello.html с таким содержанием:

```
mkdir work
cd work
touch hello.html
```

Содержание файла:

```
Hello, World
```

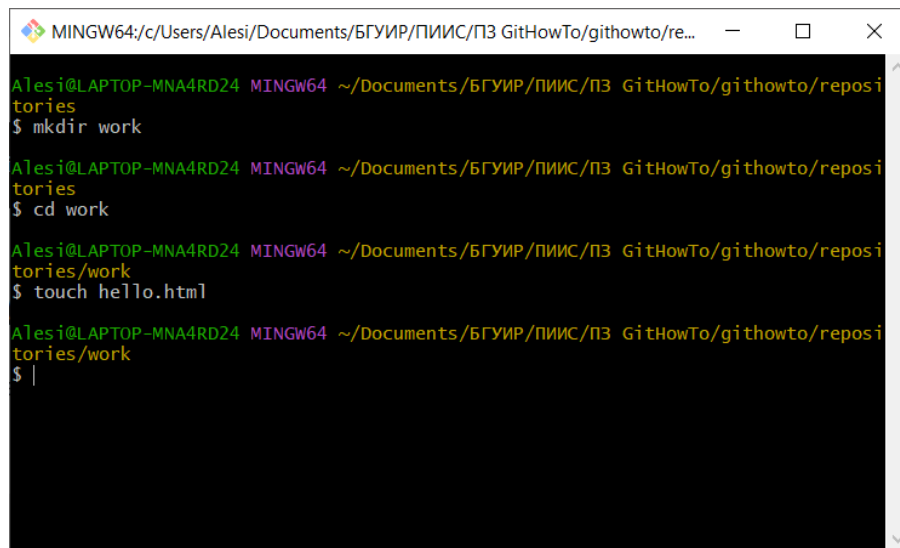


Рисунок 2 – Создание файла hello.html

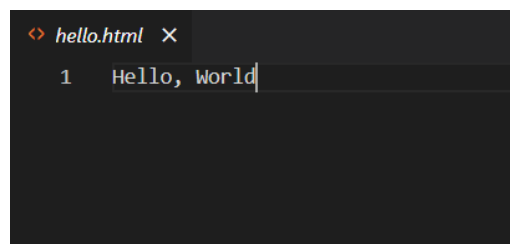


Рисунок 3 – Содержание файла hello.html

Теперь у вас есть директория с одним файлом. Чтобы создать Git-репозиторий из этой директории, выполните команду git init.

```
git init
```

Теперь давайте добавим в репозиторий страницу «Hello, World».

```
git add hello.html
git commit -m "Initial Commit"
```

```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work
$ touch hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work
$ git init
Initialized empty Git repository in C:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHo
wTo/ghithowto/repositories/work/.git/

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ git add hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ git commit -m "Initial Commit"
[main (root-commit) cc61228] Initial Commit
1 file changed, 1 insertion(+)
create mode 100644 hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$
```

Рисунок 4 – Добавление файла hello.html в Git-репозиторий

### Задание 3. Проверка состояния

Используйте команду `git status`, чтобы проверить текущее состояние репозитория.

```
git status
```

```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ git status
On branch main
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$
```

Рисунок 5 – Проверка текущего состояния репозитория

### Задание 4. Внесение изменений

Добавим кое-какие HTML-теги к нашему приветствию. Измените содержимое файла на:

```
<h1>Hello, World!</h1>
```

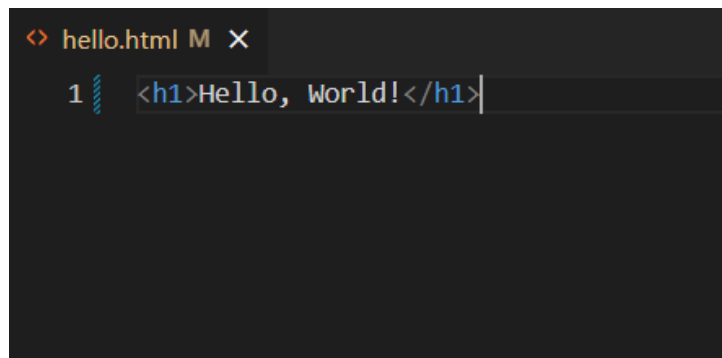


Рисунок 6 – Изменение файла hello.html

Теперь проверьте состояние рабочей директории.

```
git status
```

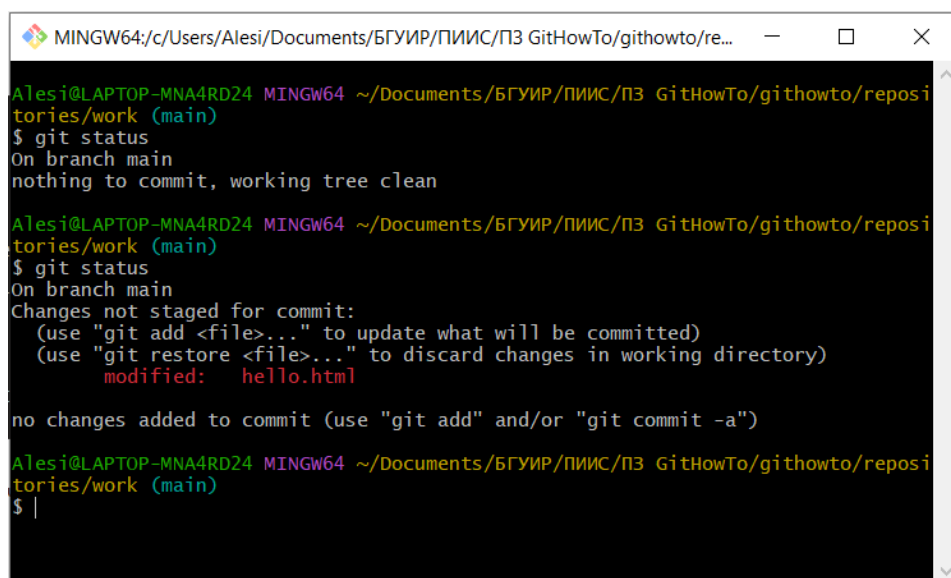


Рисунок 7 – Проверка состояния репозитория

## Задание 5. Индексация изменений

Теперь дайте команду Git проиндексировать изменения. Проверьте состояние:

```
git add hello.html  
git status
```

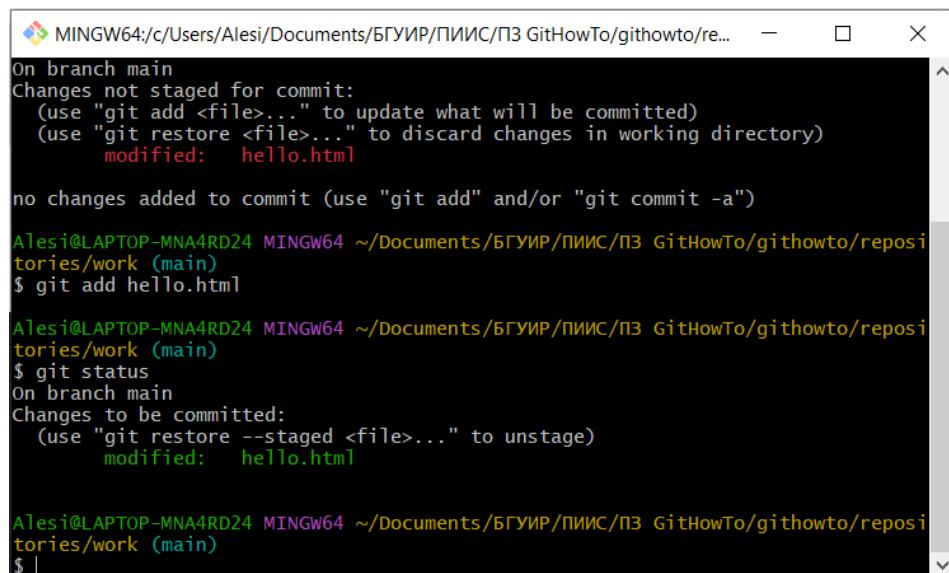
A screenshot of a terminal window with a black background and white text. The window title is 'MINGW64: c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...'. The terminal shows the following sequence of commands and output:  
On branch main  
Changes not staged for commit:  
 (use "git add <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")  
  
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi  
tories/work (main)  
\$ git add hello.html  
  
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi  
tories/work (main)  
\$ git status  
On branch main  
Changes to be committed:  
 (use "git restore --staged <file>..." to unstage)  
 modified: hello.html  
  
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi  
tories/work (main)  
\$

Рисунок 8 – Индексация изменений в репозитории

## Задание 6. Индексация и коммит

Отдельный шаг индексации в Git позволяет вам разделять большие изменения на маленькие коммиты.

Предположим, что вы отредактировали три файла (a.html, b.html, и c.html). Теперь вы хотите закоммитить все изменения, при этом чтобы изменения в a.html и b.html были одним коммитом, в то время как изменения в c.html логически не связаны с первыми двумя файлами и должны идти отдельным коммитом.

В теории, вы можете сделать следующее:

```
git add a.html  
git add b.html  
git commit -m "Changes for a and b"  
  
git add c.html  
git commit -m "Unrelated change to c"
```

Разделяя индексацию и коммит, вы имеете возможность с легкостью настроить, что идет в какой коммит.

## Задание 7. Коммит изменений

Достаточно об индексации. Давайте сделаем коммит того, что мы проиндексировали, в репозиторий.

Когда вы ранее использовали `git commit` для коммита первоначальной версии файла `hello.html` в репозиторий, вы включили метку `-m`, которая делает комментарий в командной строке. Команда `commit` позволит вам

интерактивно редактировать комментарии для коммита. Теперь давайте это проверим.

Если вы опустите метку `-m` из командной строки, Git перенесет вас в редактор по вашему выбору. Редактор выбирается из следующего списка (в порядке приоритета):

- переменная среды `GIT_EDITOR`
- параметр конфигурации `core.editor`
- переменная среды `VISUAL`
- переменная среды `EDITOR`

Сделайте коммит сейчас и проверьте состояние.

```
git commit
```

В первой строке введите комментарий: `Added h1 tag`. Сохраните файл и выйдите из редактора.

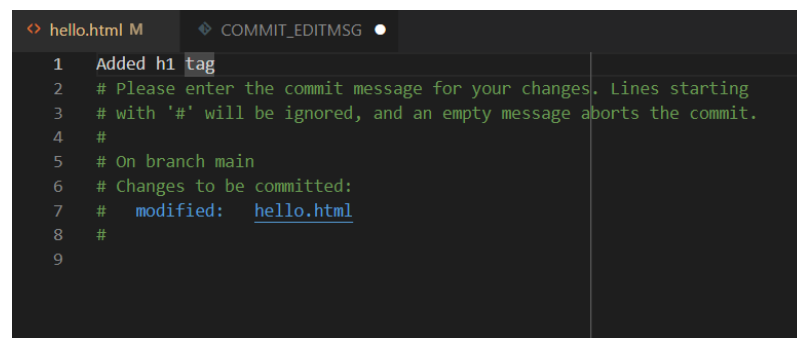


Рисунок 9 – Коммит в Visual Studio Code

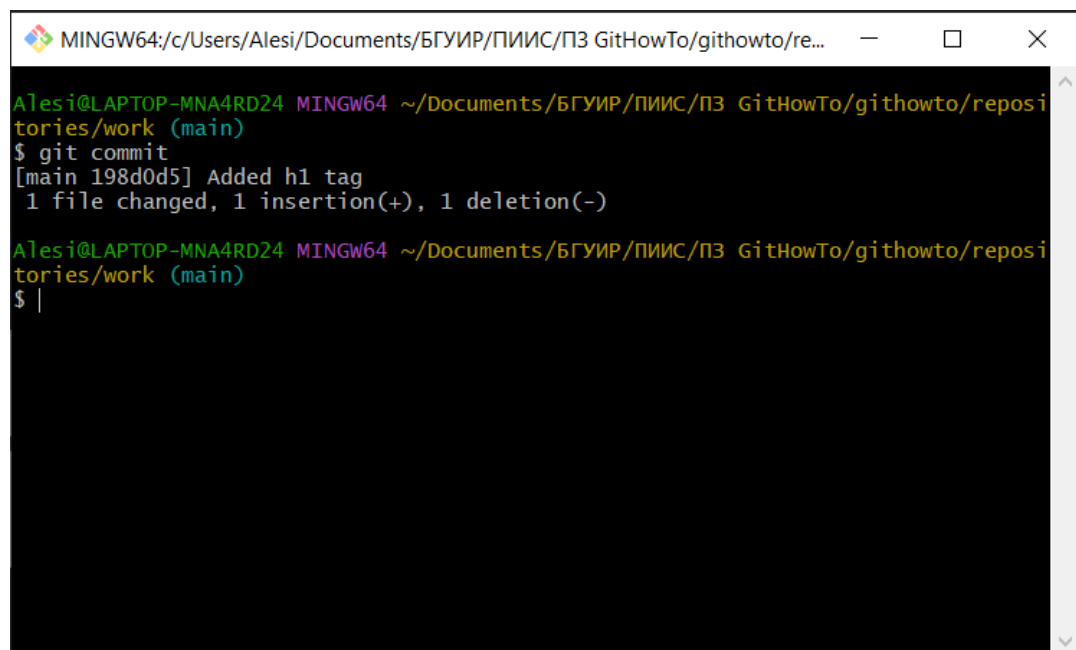
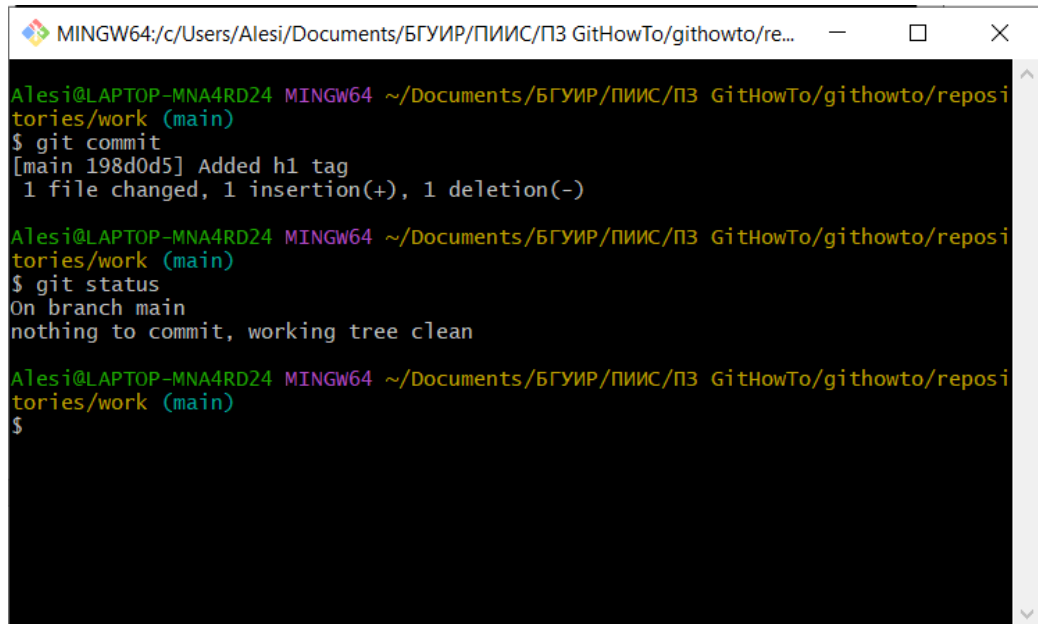


Рисунок 10 – Проверка состояния репозитория

В конце давайте еще раз проверим состояние.

```
git status
```

A screenshot of a terminal window with a black background and white text. The window title is 'MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...'. The terminal shows the following commands and output:

```
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git commit
[main 198d0d5] Added h1 tag
1 file changed, 1 insertion(+), 1 deletion(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git status
On branch main
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$
```

Рисунок 11 – Проверка состояния репозитория

Рабочая директория чиста, можем продолжить работу.

## Задание 8. Изменения, а не файлы

Большинство систем контроля версий работает с файлами: вы добавляете файл в систему, и она отслеживает изменения файла с этого момента.

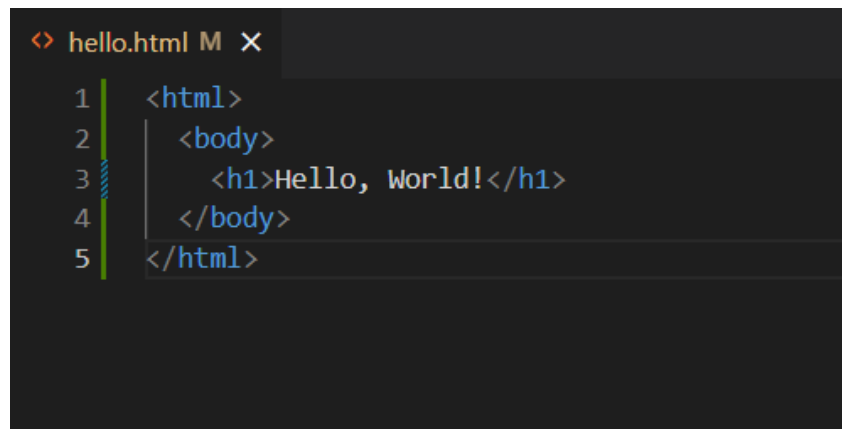
Git фокусируется на изменениях в файле, а не самом файле. Когда вы осуществляете команду `git add file`, вы не говорите Git добавить файл в репозиторий. Скорее вы говорите, что Git надо отметить текущее состояние файла, коммит которого будет произведен позже.

Мы попытаемся исследовать эту разницу в данном уроке.

Измените страницу «Hello, World», чтобы она содержала стандартные теги `<html>` и `<body>`.

```
<html>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```



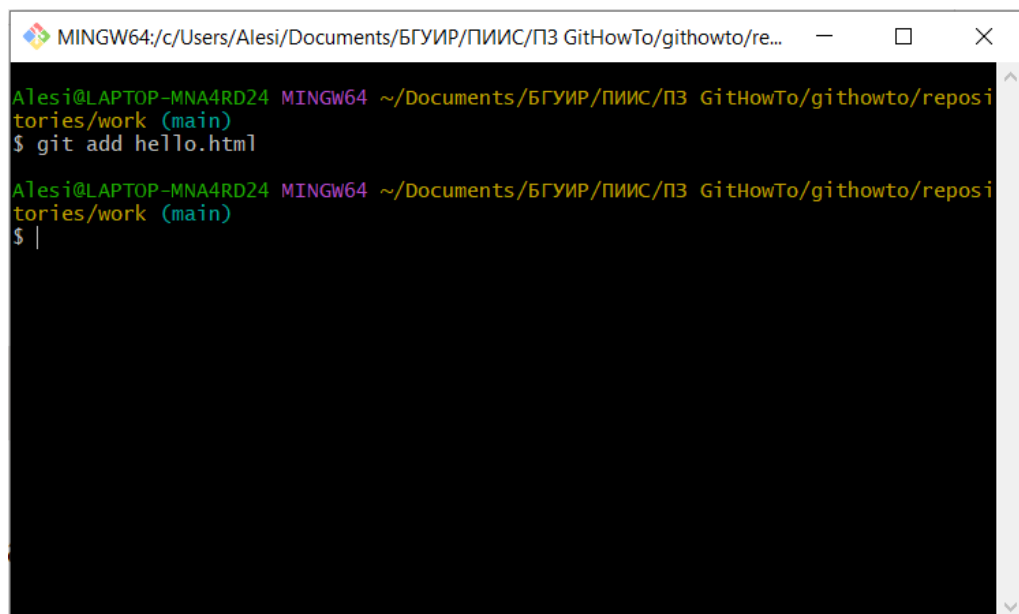
A screenshot of a code editor window titled 'hello.html M X'. The editor shows five lines of HTML code: 1. <html>, 2. <body>, 3. <h1>Hello, World!</h1>, 4. </body>, 5. </html>. The code is syntax-highlighted with blue for tags and black for text. A vertical green line is at the start of line 1, and a blue dashed line is at the start of line 3.

```
<> hello.html M X
1  <html>
2  <body>
3    <h1>Hello, World!</h1>
4  </body>
5  </html>
```

Рисунок 12 – Изменение файла hello.html

Теперь добавьте это изменение в индекс Git.

```
git add hello.html
```

A screenshot of a terminal window titled 'MINGW64: c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...'. The terminal shows the user 'Alesi@LAPTOP-MNA4RD24' in the 'MINGW64' environment at the directory '~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)'. The command '\$ git add hello.html' has been entered and executed. The prompt '\$ |' is visible on the next line.

```
MINGW64: c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git add hello.html
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ |
```

Рисунок 13 – Добавление изменения в индекс Git

Теперь добавьте заголовки HTML (секцию <head>) к странице «Hello, World».

```
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

```
<> hello.html M X
1 <html>
2   <head>
3   </head>
4   <body>
5     <h1>Hello, World!</h1>
6   </body>
7 </html>
```

Рисунок 14 – Изменение файла hello.html

Проверьте текущий статус

```
git status
```

```
MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ git add hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ git status
On branch main
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

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ |
```

Рисунок 15 – Проверка состояния репозитория

Обратите внимание на то, что hello.html указан дважды в состоянии. Первое изменение (добавление стандартных тегов) проиндексировано и готово к коммиту. Второе изменение (добавление заголовков HTML) является непроиндексированным. Если бы вы делали коммит сейчас, заголовки не были бы сохранены в репозиторий.

Давайте проверим.

Произведите коммит проиндексированного изменения (значение по умолчанию), а затем еще раз проверьте состояние.

```
git commit -m "Added standard HTML page tags"
git status
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
(use "git restore <file>..." to discard changes in working directory)
modified:   hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git commit -m "Added standart HTML page tags"
[main cd2666d] Added standart HTML page tags
1 file changed, 5 insertions(+), 1 deletion(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git status
On branch main
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")

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ |
```

Рисунок 16 – Коммит и проверка состояния репозитория

Команда status показывает, что в файле hello.html ещё есть незаписанные изменения, но область подготовки уже пуста.

Теперь добавьте второе изменение в индекс, а затем проверьте состояние с помощью команды git status.

```
git add .
git status
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
On branch main
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")

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git add .

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ |
```

Рисунок 17 – Добавление второго изменения в индекс и проверка состояния репозитория

Второе изменение было проиндексировано и готово к коммиту. Сделайте коммит второго изменения.

```
git commit -m "Added HTML header"
```

```
MINGW64:/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/re...
no changes added to commit (use "git add" and/or "git commit -a")

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git add .

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git commit -m "Added HTML header"
[main ea06feb] Added HTML header
1 file changed, 2 insertions(+)

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ |
```

Рисунок 18 – Коммит второго изменения

## Задание 9. История

Получение списка произведенных изменений – функция команды git log.

git log

```
MINGW64:/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git log
commit ea06febfe5df62fc347b8b346e6d77876ba21a5f (HEAD -> main)
Author: Alesia <Alesia.Loika@gmail.com>
Date:   Fri Nov 15 20:34:22 2024 +0300

    Added HTML header

commit cd2666d0feaadd5f807730cec536315db4393935
Author: Alesia <Alesia.Loika@gmail.com>
Date:   Fri Nov 15 20:30:12 2024 +0300

    Added standart HTML page tags

commit 198d0d59bbeff765603a5a702fb771be2028c3fe
Author: Alesia <Alesia.Loika@gmail.com>
Date:   Fri Nov 15 20:09:12 2024 +0300

    Added h1 tag

commit cc61228513374708018b6e73405b6f19ebc8270b
Author: Alesia <Alesia.Loika@gmail.com>
Date:   Fri Nov 15 19:48:02 2024 +0300

    Initial Commit

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work (main)
$ |
```

Рисунок 19 – Получение списка изменений

Вот список всех четырех коммитов в репозиторий, которые мы успели совершить.

Вы полностью контролируете то, что отображает log. Например, однострочный формат:

```
git log --pretty=oneline
```

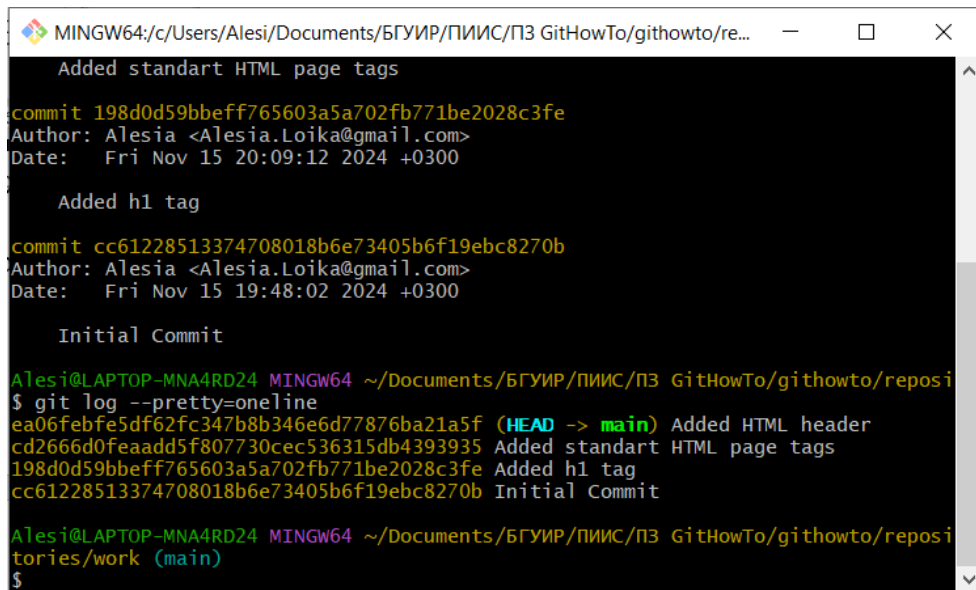
A screenshot of a terminal window with a black background and white text. The window title is 'MINGW64: c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...'. The terminal shows the output of 'git log --pretty=oneline'. It lists three commits in a single-line format: 'commit 198d0d59bbeff765603a5a702fb771be2028c3fe' with author 'Alesia <Alesia.Loika@gmail.com>' and date 'Fri Nov 15 20:09:12 2024 +0300', followed by 'Added standart HTML page tags'. The second commit is 'commit cc61228513374708018b6e73405b6f19ebc8270b' with the same author and date 'Fri Nov 15 19:48:02 2024 +0300', followed by 'Added h1 tag'. The third commit is 'Initial Commit'. Below the log, the prompt 'Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi' is shown, followed by '\$ git log --pretty=oneline' and its output. At the bottom, the prompt 'Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi' is shown again, followed by 'tories/work (main)' and '\$'.

Рисунок 20 – Получение списка изменений в однострочном формате

Вот еще интересные варианты просмотра истории:

```
git log --oneline --max-count=2
git log --oneline --since="5 minutes ago"
git log --oneline --until="5 minutes ago"
git log --oneline --author="Your Name"
git log --oneline -all
```

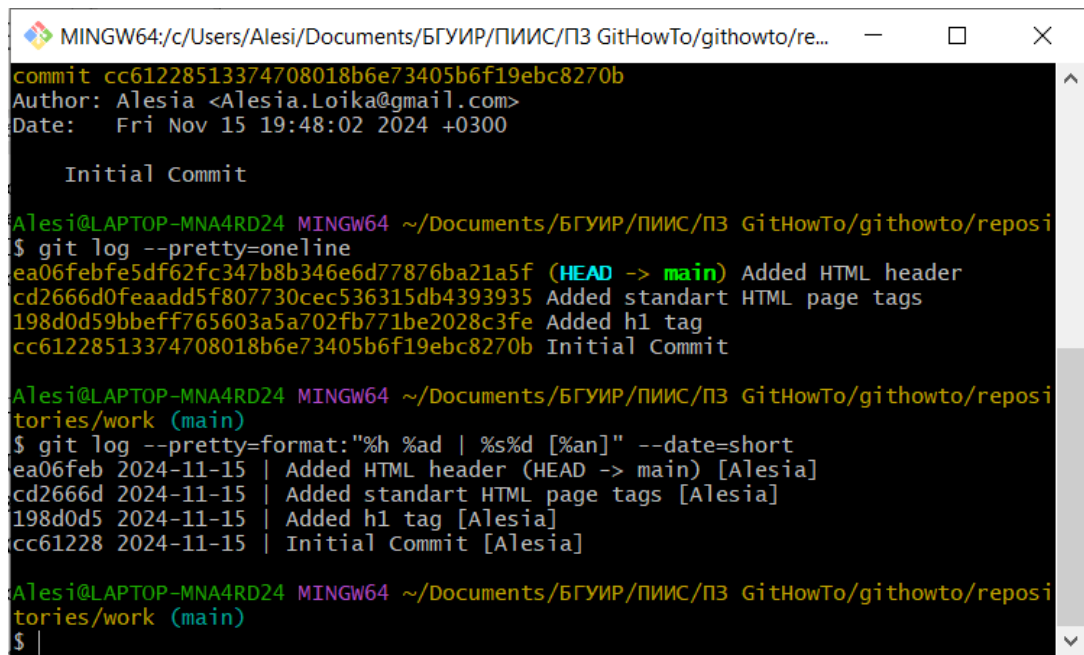
Существует огромное количество вариантов просмотра истории, вы можете порыться на странице руководства git-log, чтобы увидеть их все.

Вот что я использую для просмотра изменений, сделанных за последнюю неделю. Я добавлю --author=Alexander, если я хочу увидеть только изменения, которые сделал я.

```
git log --all --pretty=format:"%h %cd %s (%an)" --since="7 days ago"
```

Со временем, я решил, что для большей части моей работы мне подходит следующий формат лога.

```
git log --pretty=format:"%h %ad | %s%d [%an]" --date=short
```

A screenshot of a Windows terminal window with a black background and white text. The window title is 'MINGW64:/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/re...'. The terminal shows the output of 'git log --pretty=oneline' and 'git log --pretty=format: "%h %ad | %s%d [%an]" --date=short'. The first command shows four commits with their hashes and messages. The second command shows the same commits in a structured format with columns for hash, date, message, and author. The user is 'Alesia' and the repository is '~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work (main)'.

```
commit cc61228513374708018b6e73405b6f19ebc8270b
Author: Alesia <Alesia.Loika@gmail.com>
Date:   Fri Nov 15 19:48:02 2024 +0300

Initial Commit

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
$ git log --pretty=oneline
ea06febfe5df62fc347b8b346e6d77876ba21a5f (HEAD -> main) Added HTML header
cd2666d0feaadd5f807730cec536315db4393935 Added standart HTML page tags
198d0d59bbeff765603a5a702fb771be2028c3fe Added h1 tag
cc61228513374708018b6e73405b6f19ebc8270b Initial Commit

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git log --pretty=format: "%h %ad | %s%d [%an]" --date=short
ea06feb 2024-11-15 | Added HTML header (HEAD -> main) [Alesia]
cd2666d 2024-11-15 | Added standart HTML page tags [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$
```

Рисунок 21 – Получение списка изменений в подходящем формате

Давайте рассмотрим его в деталях:

--pretty="..." — определяет формат вывода.

%h — укороченный хеш коммита.

%ad — дата коммита.

| — просто визуальный разделитель.

%s — комментарий.

%d — дополнения коммита («головы» веток или теги).

%an — имя автора.

--date=short — сохраняет формат даты коротким и симпатичным.

Таким образом, каждый раз, когда вы захотите посмотреть лог, вам придется много печатать. К счастью, существует несколько опций конфигурации Git, позволяющих настроить формат вывода истории по умолчанию:

```
git config --global format.pretty '%h %ad | %s%d [%an]'
```

```
git config --global log.date short
```

```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
ea06febfe5df62fc347b8b346e6d77876ba21a5f (HEAD -> main) Added HTML header
cd2666d0feaadd5f807730cec536315db4393935 Added standart HTML page tags
198d0d59bbeff765603a5a702fb771be2028c3fe Added h1 tag
cc61228513374708018b6e73405b6f19ebc8270b Initial Commit

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log --pretty=format:"%h %ad | %s%d [%an]" --date=short
ea06feb 2024-11-15 | Added HTML header (HEAD -> main) [Alesia]
cd2666d 2024-11-15 | Added standart HTML page tags [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git config --global format.pretty '%h %ad | %s%d [%an]'
```

Рисунок 22 – Настройка формата вывода истории по умолчанию

Оба gitx (для Mac) и gitk (для любой платформы) полезны в изучении истории изменений.

## Задание 10. Получение старых версий

Git позволяет очень просто путешествовать во времени, по крайней мере, для вашего проекта. Команда checkout обновит вашу рабочую директорию до любого предыдущего коммита.

Получите хеши предыдущих коммитов:

```
git log
```

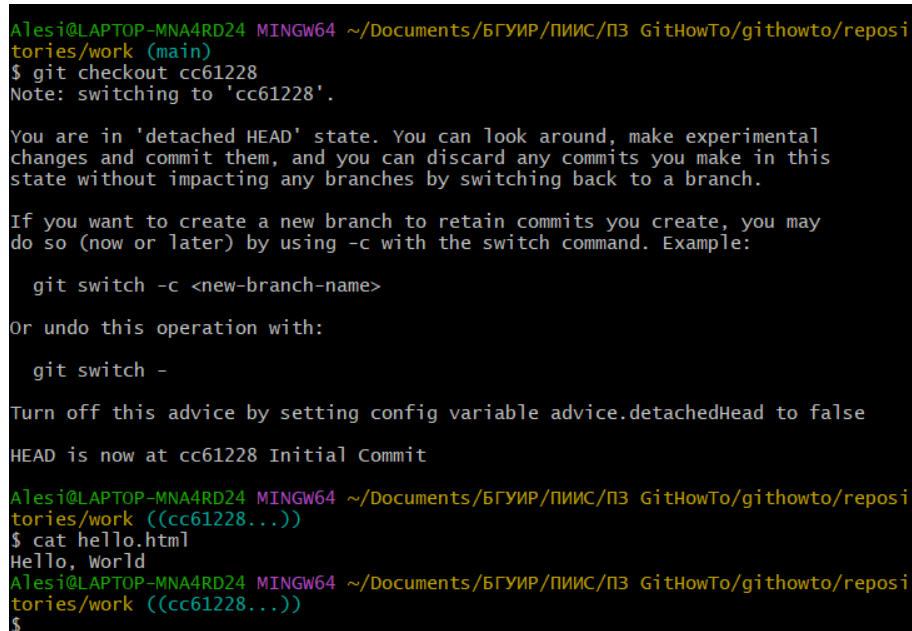
```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log
ea06feb 2024-11-15 | Added HTML header (HEAD -> main) [Alesia]
cd2666d 2024-11-15 | Added standart HTML page tags [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 23 – Получение хешей предыдущих коммитов

Просмотрите историю изменений и найдите хеш первого коммита. Он должен быть в последней строке результата `git log`. Используйте этот хеш (достаточно первых 7 символов) в команде ниже. Затем проверьте содержимое файла `hello.html`.

```
git checkout <hash>
cat hello.html
```



```
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git checkout cc61228
Note: switching to 'cc61228'.

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>

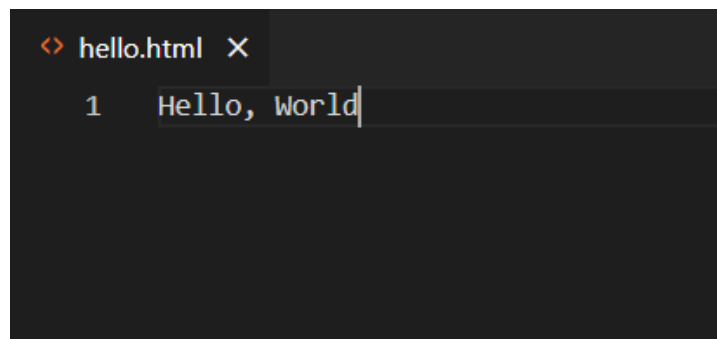
Or undo this operation with:

    git switch -

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

HEAD is now at cc61228 Initial Commit
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((cc61228...))
$ cat hello.html
Hello, world
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((cc61228...))
$
```

Рисунок 24 – Возвращение к первой версии файла `hello.html`



```
<> hello.html X
1 Hello, world
```

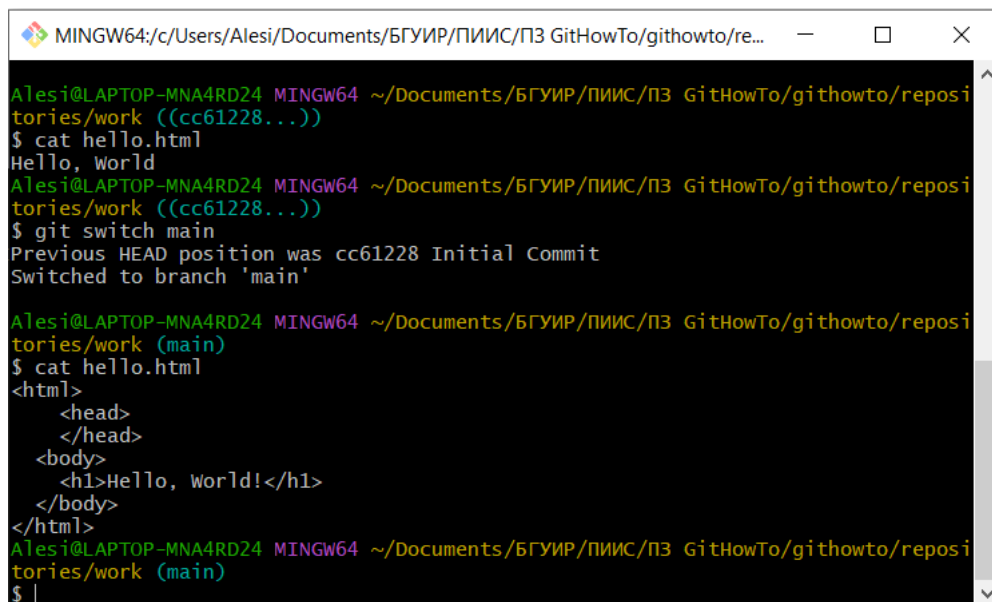
Рисунок 25 – Содержание файла `hello.html`

Обратите внимание, что сейчас содержимое файла `hello.html` — это тот самый текст, с которого мы начинали.

Чтобы вернуться к последней версии нашего кода, нам нужно переключиться на ветку по умолчанию, `main`. Для переключения между ветками можно воспользоваться командой `switch`.

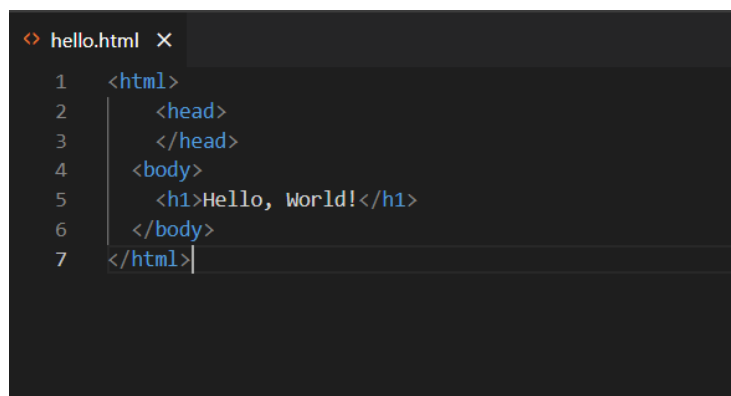
```
git switch main
cat hello.html
```





```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ cat hello.html
Hello, World
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ git switch main
Previous HEAD position was cc61228 Initial Commit
Switched to branch 'main'
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ cat hello.html
<html>
  <head>
</head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

Рисунок 26 – Возвращение к последней версии в ветке main



```
hello.html x
1 <html>
2   <head>
3   </head>
4   <body>
5     <h1>Hello, World!</h1>
6   </body>
7 </html>
```

Рисунок 27 – Содержание файла hello.html

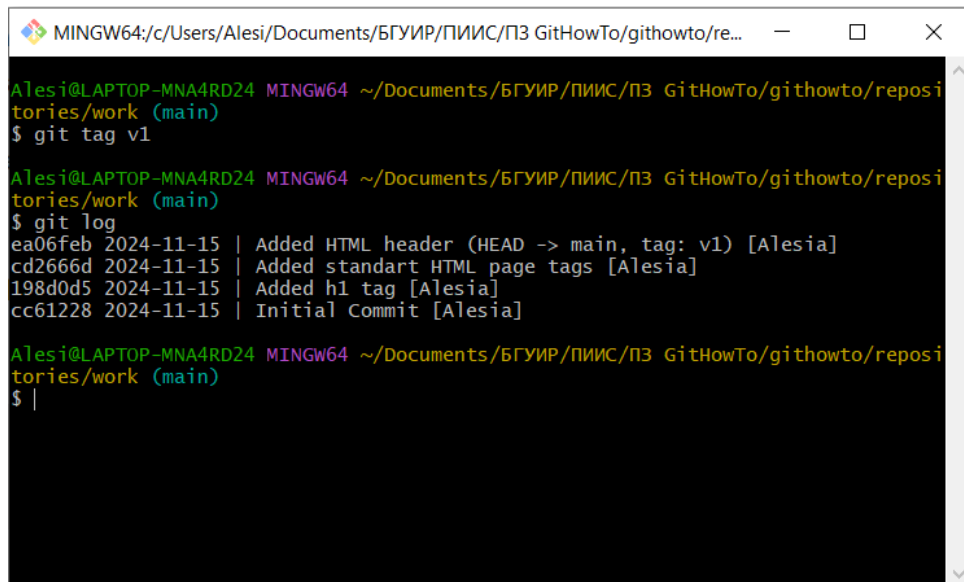
Здесь main — имя ветки по умолчанию. Переключаясь на ветку, вы попадаете на её последнюю версию.

## Задание 11. Создание тегов версий

Думаю, вы согласитесь, что работать с хешами коммитов напрямую просто неудобно. Разве не было бы здорово, если бы вы могли обозначать конкретные коммиты понятными для человека названиями? Таким образом, вы могли бы четко видеть важные вехи в истории проекта. Кроме того, вы могли бы легко перейти к определенной версии проекта по ее названию. Именно для этого в Git придумали теги.

Давайте назовем текущую версию страницы hello.html первой, то есть v1. Создайте тег первой версии:

```
git tag v1
git log
```

A screenshot of a terminal window with a black background and green text. The window title is "MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...". The user is at the prompt "Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)". They enter the command "\$ git tag v1". The prompt changes to "(main)". They then enter "\$ git log", which shows a list of commits: "ea06feb 2024-11-15 | Added HTML header (HEAD -> main, tag: v1) [Alesia]", "cd2666d 2024-11-15 | Added standart HTML page tags [Alesia]", "198d0d5 2024-11-15 | Added h1 tag [Alesia]", and "cc61228 2024-11-15 | Initial Commit [Alesia]". The prompt returns to "\$ |".

```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git tag v1

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git log
ea06feb 2024-11-15 | Added HTML header (HEAD -> main, tag: v1) [Alesia]
cd2666d 2024-11-15 | Added standart HTML page tags [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]


Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ |
```

Рисунок 28 – Создание тега первой версии

Теперь текущая версия страницы называется v1.

Обозначим версию, предшествующую текущей, названием v1-beta. Прежде всего, мы переключимся на предыдущую версию. Вместо того чтобы искать хеш коммита, мы будем использовать обозначение ^, а именно v1^, указывающее на коммит, предшествующий v1.

```
git checkout v1^
cat hello.html
```

A screenshot of a terminal window with a black background and green text. The window title is "MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...". The user is at the prompt "Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)". They enter the command "\$ git checkout v1^". The prompt changes to "((cd2666d...))". They then enter "\$ cat hello.html", which shows the content of the file: "<html>\n <body>\n <h1>Hello, World!</h1>\n </body>\n</html>". The prompt returns to "\$ |".

```
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git checkout v1^
Note: switching to 'v1^'.

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 cd2666d Added standart HTML page tags

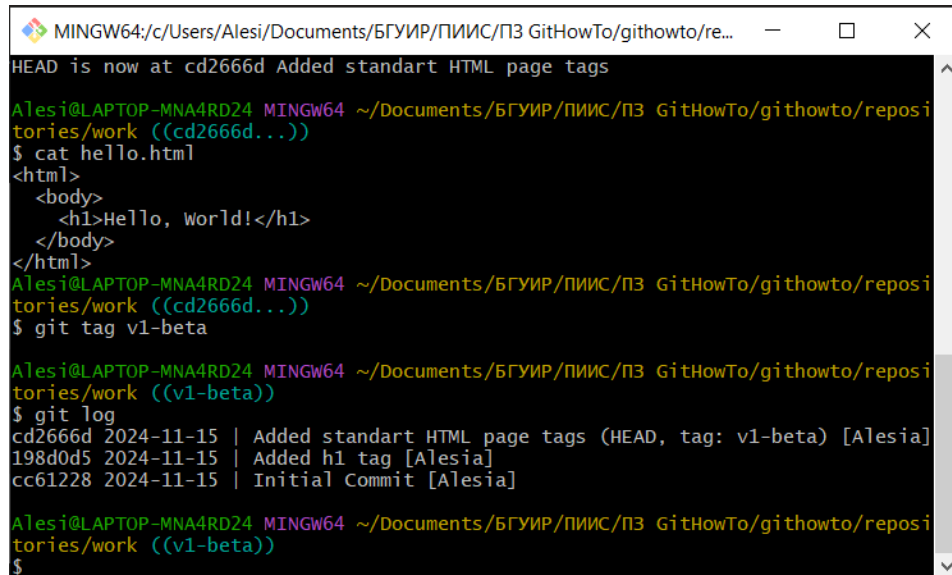
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((cd2666d...))
$ cat hello.html
<html>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((cd2666d...))
$ |
```

Рисунок 29 – Переключение на предыдущую версию

Это версия с тегами <html> и <body>, но еще пока без <head>. Давайте сделаем ее версией v1-beta.

```
git tag v1-beta
git log
```



```
MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
HEAD is now at cd2666d Added standart HTML page tags
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((cd2666d...))
$ cat hello.html
<html>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((cd2666d...))
$ git tag v1-beta

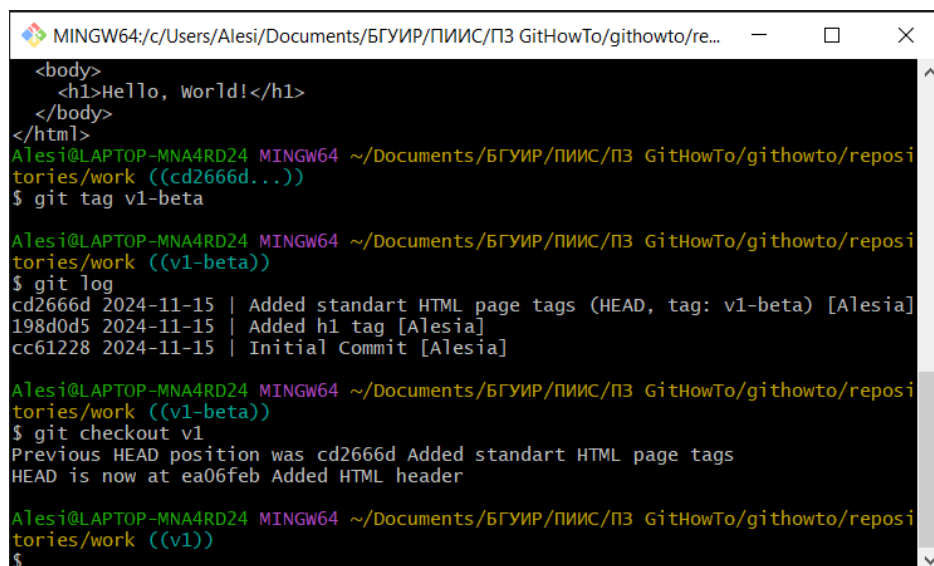
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git log
cd2666d 2024-11-15 | Added standart HTML page tags (HEAD, tag: v1-beta) [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$
```

Рисунок 30 – Обозначение версии v1-beta

Теперь попробуйте переключаться между двумя отмеченными версиями.

```
git checkout v1
git checkout v1-beta
```



```
MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((cd2666d...))
$ git tag v1-beta

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git log
cd2666d 2024-11-15 | Added standart HTML page tags (HEAD, tag: v1-beta) [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git checkout v1
Previous HEAD position was cd2666d Added standart HTML page tags
HEAD is now at ea06feb Added HTML header

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1))
$
```

Рисунок 31 – Переключение на версию v1

```
<> hello.html X
1  <html>
2    <head>
3  </head>
4    <body>
5      <h1>Hello, World!</h1>
6    </body>
7  </html>
```

Рисунок 32 – Файл hello.html версии v1

```
MINGW64:/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
$ git tag v1-beta
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((v1-beta))
$ git log
cd2666d 2024-11-15 | Added standart HTML page tags (HEAD, tag: v1-beta) [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((v1-beta))
$ git checkout v1
Previous HEAD position was cd2666d Added standart HTML page tags
HEAD is now at ea06feb Added HTML header

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((v1))
$ git checkout v1-beta
Previous HEAD position was ea06feb Added HTML header
HEAD is now at cd2666d Added standart HTML page tags

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((v1-beta))
$
```

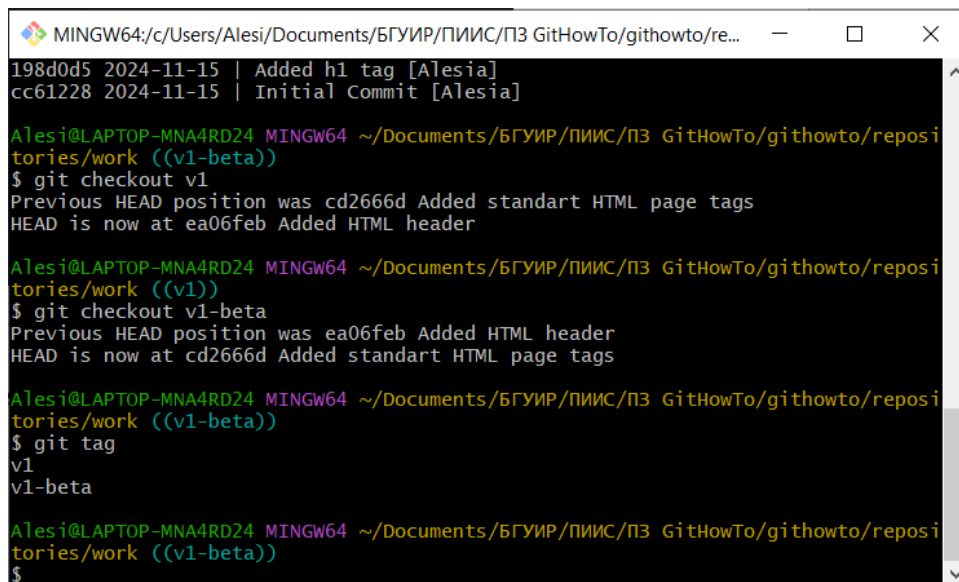
Рисунок 33 – Переключение на версию v1-beta

```
<> hello.html X
1  <html>
2    <body>
3      <h1>Hello, World!</h1>
4    </body>
5  </html>
```

Рисунок 34 – Файл hello.html версии v1-beta

Вы можете увидеть, какие теги доступны, используя команду git tag.

```
git tag
```



```
MINGW64:/c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git checkout v1
Previous HEAD position was cd2666d Added standart HTML page tags
HEAD is now at ea06feb Added HTML header

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1))
$ git checkout v1-beta
Previous HEAD position was ea06feb Added HTML header
HEAD is now at cd2666d Added standart HTML page tags

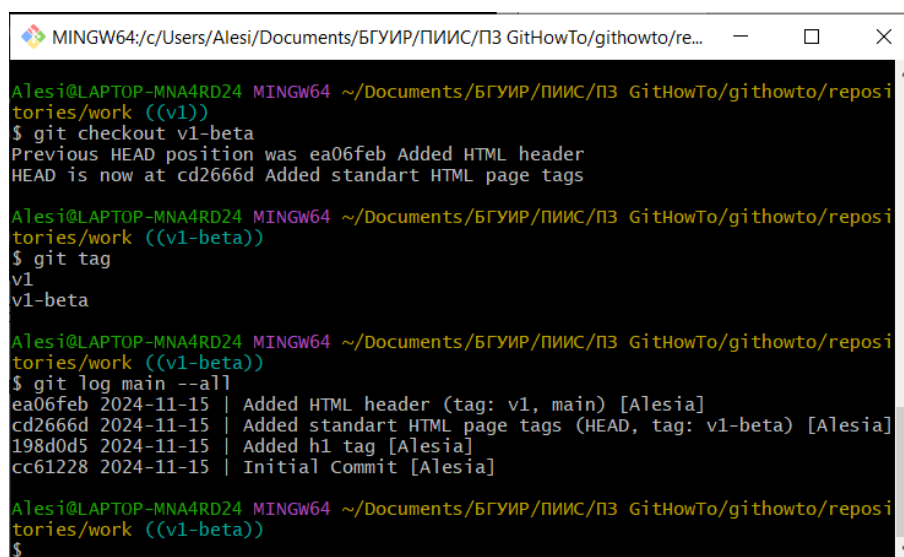
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git tag
v1
v1-beta

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$
```

Рисунок 35 – Просмотр доступных тегов

Вы также можете посмотреть теги в логе.

```
git log main --all
```



```
MINGW64:/c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1))
$ git checkout v1-beta
Previous HEAD position was ea06feb Added HTML header
HEAD is now at cd2666d Added standart HTML page tags

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git tag
v1
v1-beta

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$ git log main --all
ea06feb 2024-11-15 | Added HTML header (tag: v1, main) [Alesia]
cd2666d 2024-11-15 | Added standart HTML page tags (HEAD, tag: v1-beta) [Alesia]
198d0d5 2024-11-15 | Added h1 tag [Alesia]
cc61228 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work ((v1-beta))
$
```

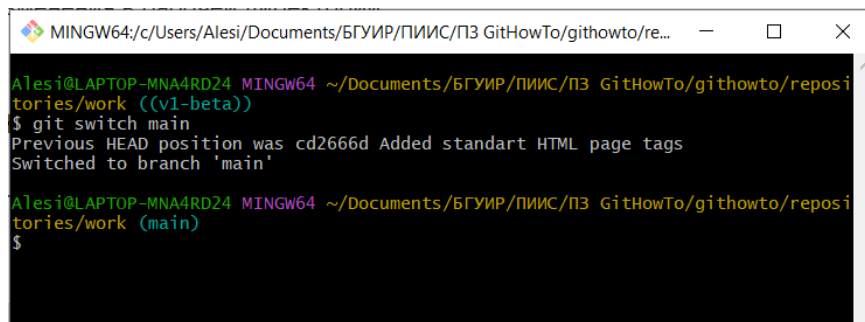
Рисунок 36 – Просмотр тегов в логе

Вы можете видеть теги (v1 и v1-beta) в логе вместе с именем ветки (main). Кроме того, метка HEAD показывает коммит, на который вы переключились (на данный момент это v1-beta).

## Задание 12. Отмена локальных изменений (до индексации)

Убедитесь, что вы находитесь на последнем коммите ветки main, прежде чем продолжить работу.

```
git switch main
```



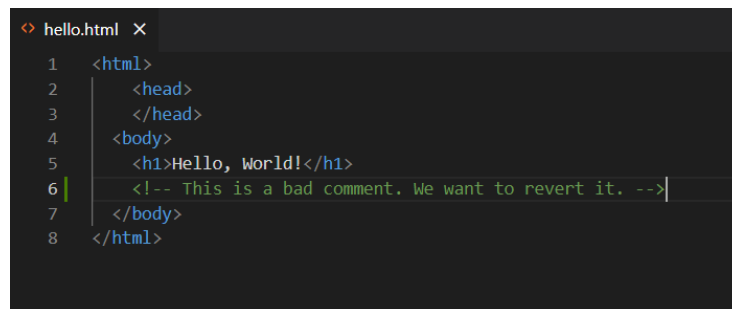
```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((v1-beta))
$ git switch main
Previous HEAD position was cd2666d Added standart HTML page tags
Switched to branch 'main'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$
```

Рисунок 37 – Проверка, что мы на последнем коммите ветки main

Иногда после того как вы изменили файл в рабочей директории, вы передумали и хотите просто вернуться к тому, что уже было закоммичено. Команда checkout справится с этой задачей.

Внесите изменение в файл hello.html в виде нежелательного комментария.

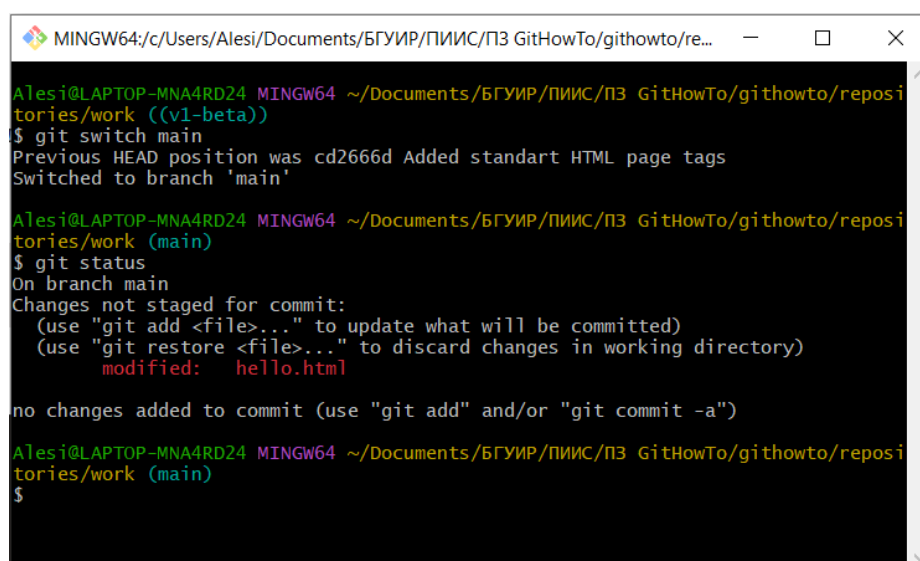


```
hello.html x
1 <html>
2   <head>
3   </head>
4   <body>
5     <h1>Hello, World!</h1>
6     <!-- This is a bad comment. We want to revert it. -->
7   </body>
8 </html>
```

Рисунок 38 – Изменение файла hello.html

Сначала проверьте состояние рабочей директории.

```
git status
```



```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work ((v1-beta))
$ git switch main
Previous HEAD position was cd2666d Added standart HTML page tags
Switched to branch 'main'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git status
On branch main
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")

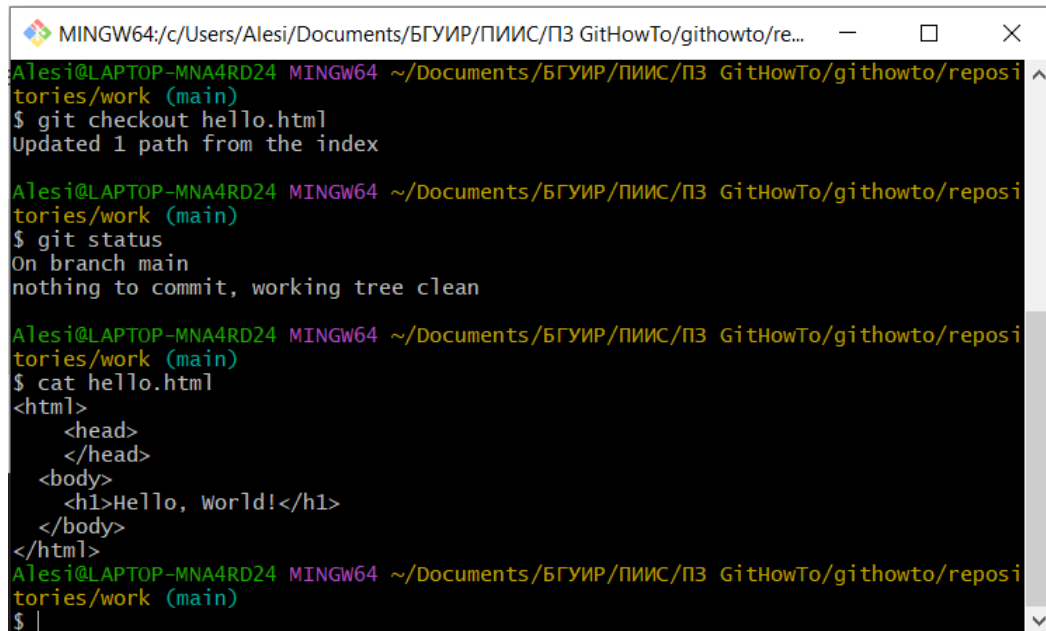
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$
```

Рисунок 39 – Проверка состояния рабочей директории

Мы видим, что файл `hello.html` был изменен, но еще не проиндексирован.

Используйте команду `checkout` для переключения в версию файла `hello.html` в репозитории.

```
git checkout hello.html
git status
cat hello.html
```

A screenshot of a terminal window titled "MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/re...". The terminal shows the following commands and output:

```
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repositories/work (main)
$ git checkout hello.html
Updated 1 path from the index

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repositories/work (main)
$ git status
On branch main
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repositories/work (main)
$ cat hello.html
<html>
  <head>
  </head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repositories/work (main)
$
```

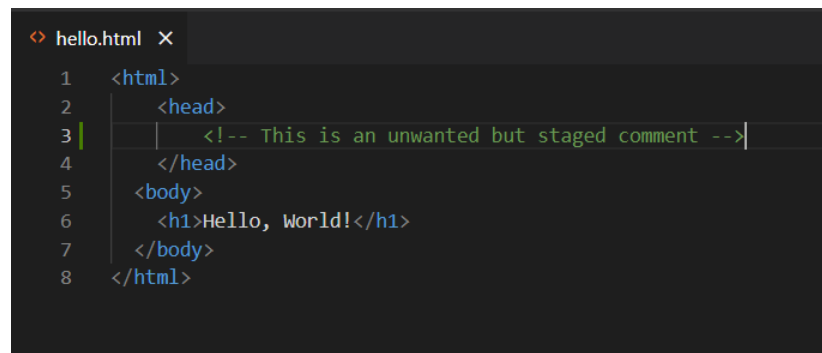
Рисунок 40 – Переключение в версию файла `hello.html` в репозитории

Команда `status` показывает нам, что в рабочей директории не было сделано никаких незафиксированных изменений. И «нежелательный комментарий» больше не является частью содержимого файла.

### Задание 13. Отмена проиндексированных изменений (перед коммитом)

Внесите изменение в файл `hello.html` в виде нежелательного комментария:

```
<html>
  <head>
    <!-- This is an unwanted but staged comment -->
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```



```
<> hello.html x
1 <html>
2   <head>
3     <!-- This is an unwanted but staged comment -->
4   </head>
5   <body>
6     <h1>Hello, World!</h1>
7   </body>
8 </html>
```

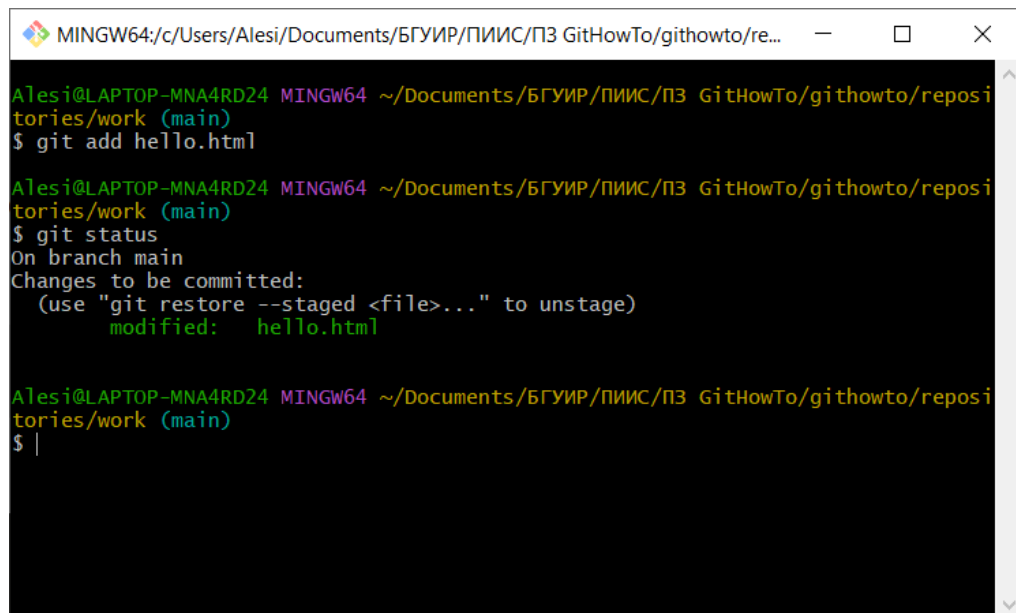
Рисунок 41 – Изменение файла hello.html

Проиндексируйте это изменение.

```
git add hello.html
```

Проверьте состояние нежелательного изменения.

```
git status
```



```
MINGW64:/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git add hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ |
```

Рисунок 42 – Индексирование изменения и проверка состояния

Состояние показывает, что изменение было проиндексировано и готово к коммиту.

Команда reset сбрасывает область подготовки к HEAD. Это очищает область подготовки от изменений, которые мы только что проиндексировали.

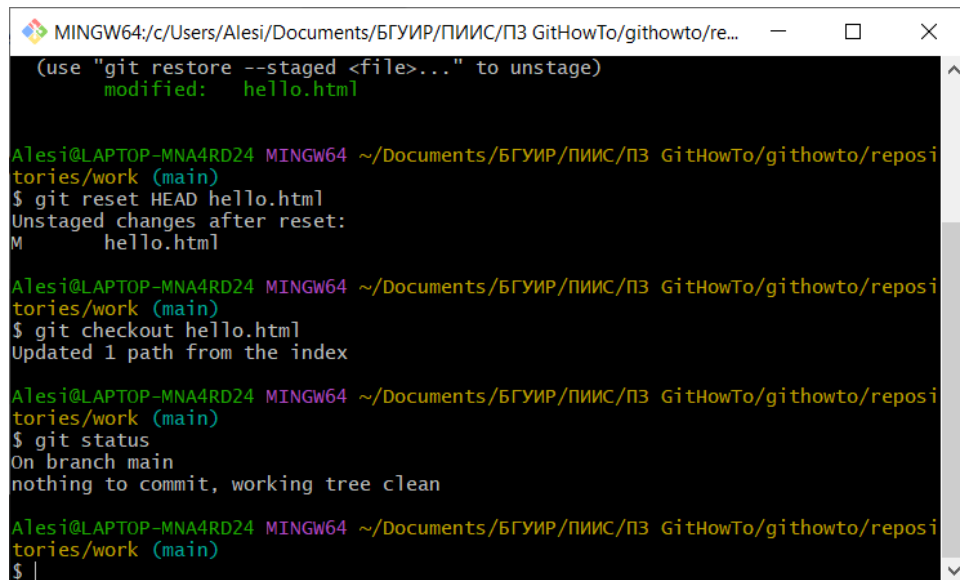
```
git reset HEAD hello.html
```

Команда reset (по умолчанию) не изменяет рабочую директорию. Поэтому рабочая директория всё еще содержит нежелательный комментарий. Мы можем использовать команду checkout из предыдущего урока, чтобы убрать нежелательные изменения в рабочей директории.



Переключитесь на версию коммита:

```
git checkout hello.html
git status
```



```
MINGW64:/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
(use "git restore --staged <file>..." to unstage)
modified:   hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ git reset HEAD hello.html
Unstaged changes after reset:
M    hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ git checkout hello.html
Updated 1 path from the index

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ git status
On branch main
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ |
```

Рисунок 43 – Сброс области подготовки и убирание нежелательного комментария

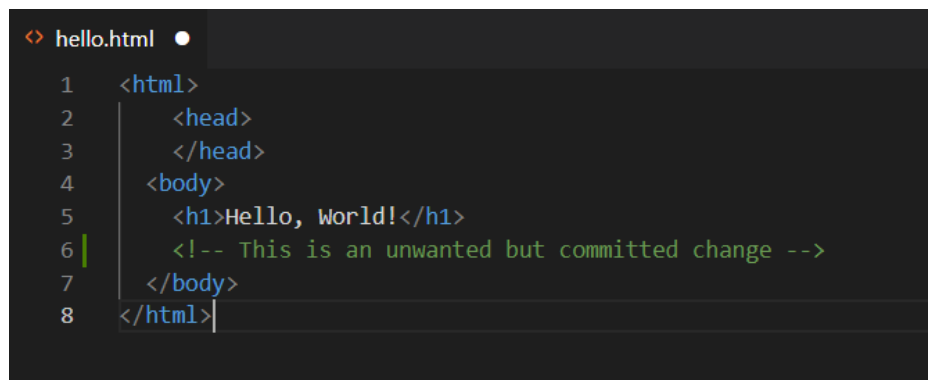
Наша рабочая директория опять чиста.

## Задание 14. Отмена коммитов

Иногда вы понимаете, что новые коммиты являются неверными, и хотите их отменить. Есть несколько способов решения этого вопроса, здесь мы будем использовать самый безопасный.

Мы отменим коммит путем создания нового коммита, отменяющего нежелательные изменения.

Измените файл `hello.html` на следующий.



```
<> hello.html ●
1  <html>
2    <head>
3    </head>
4    <body>
5      <h1>Hello, World!</h1>
6      <!-- This is an unwanted but committed change -->
7    </body>
8  </html>
```

Рисунок 44 – Изменение файла `hello.html`

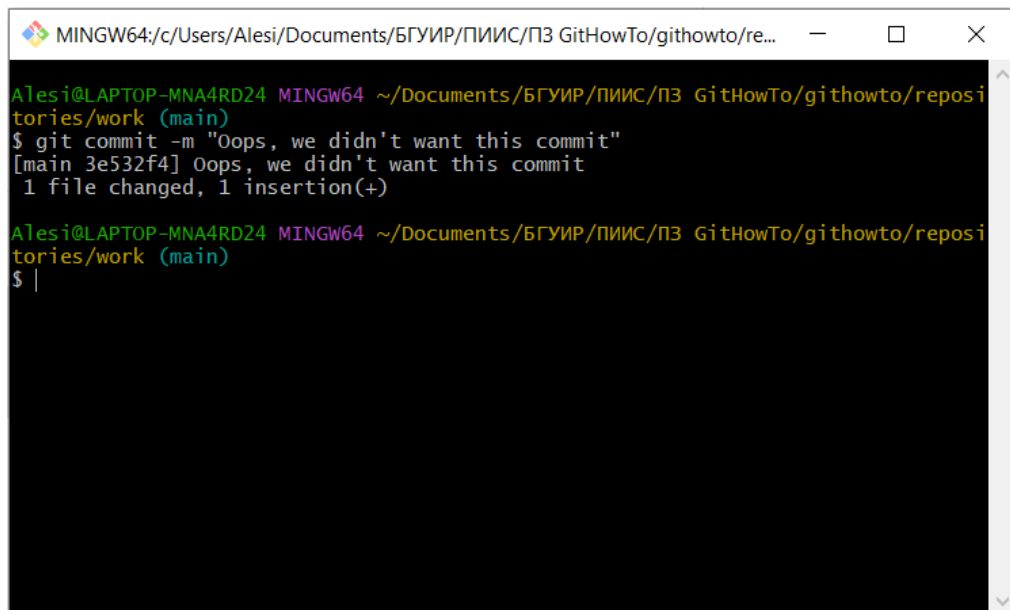
Затем выполните команды:

```
git add hello.html
git commit -m "Oops, we didn't want this commit"
```

Чтобы отменить коммит, нам необходимо сделать коммит, который удаляет изменения, сохраненные нежелательным коммитом.

```
git revert HEAD
```

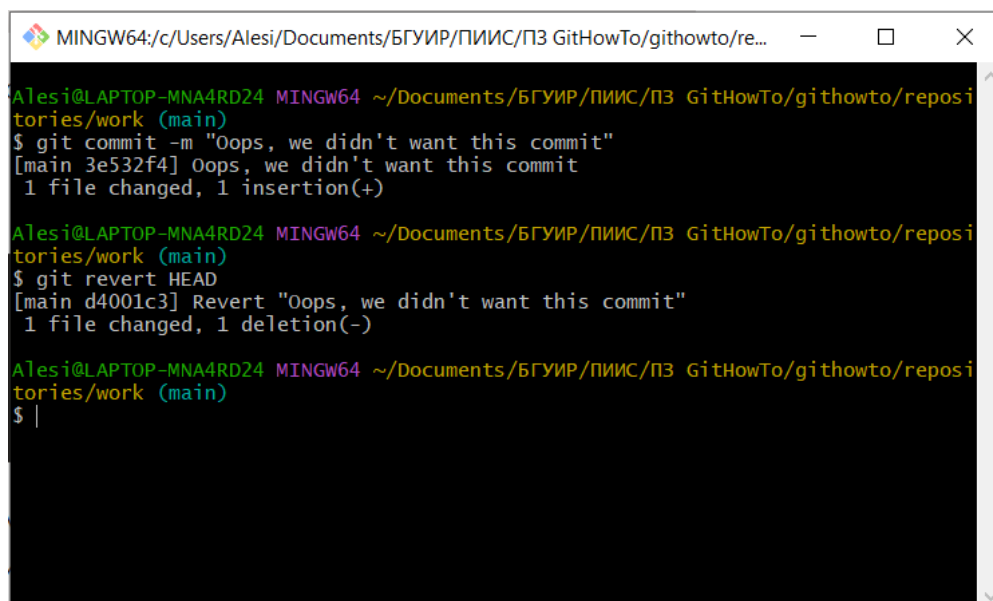
Перейдите в редактор, где вы можете отредактировать коммит-сообщение по умолчанию или оставить все как есть.



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git commit -m "Oops, we didn't want this commit"
[main 3e532f4] Oops, we didn't want this commit
1 file changed, 1 insertion(+)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ |
```

Рисунок 45 – Добавление нежелательного коммита



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git commit -m "Oops, we didn't want this commit"
[main 3e532f4] Oops, we didn't want this commit
1 file changed, 1 insertion(+)

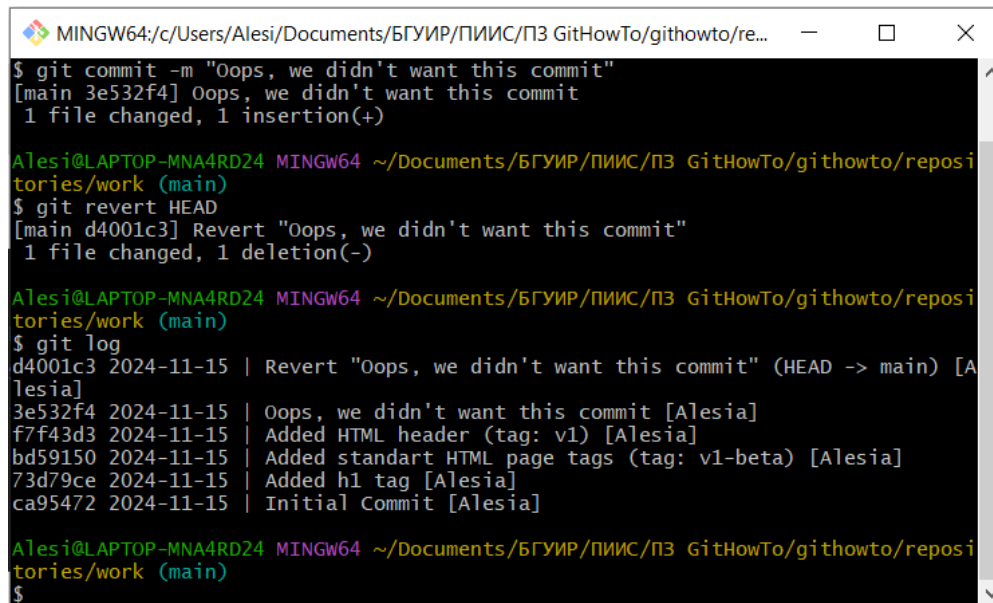
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git revert HEAD
[main d4001c3] Revert "Oops, we didn't want this commit"
1 file changed, 1 deletion(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ |
```

Рисунок 46 – Удаление нежелательного коммита

Так как мы отменили самый последний произведенный коммит, мы смогли использовать метку HEAD в качестве аргумента для отмены коммита. Мы можем отменить любой произвольный коммит в истории, указав его хеш.

Проверка лога показывает нежелательные и отмененные коммиты в наш репозиторий.



```
MINGW64/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
$ git commit -m "Oops, we didn't want this commit"
[main 3e532f4] Oops, we didn't want this commit
1 file changed, 1 insertion(+)

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git revert HEAD
[main d4001c3] Revert "Oops, we didn't want this commit"
1 file changed, 1 deletion(-)

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log
d4001c3 2024-11-15 | Revert "Oops, we didn't want this commit" (HEAD -> main) [Alesia]
3e532f4 2024-11-15 | Oops, we didn't want this commit [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 47 – Проверка лога

Эта техника будет работать с любым коммитом (хотя, возможно, возникнут конфликты). Она безопасна в использовании даже в публичных ветках удаленных репозиториях.

Далее давайте посмотрим на технику, которая может быть использована для удаления последних коммитов из истории репозитория.

### Задание 15. Удаление коммитов из ветки

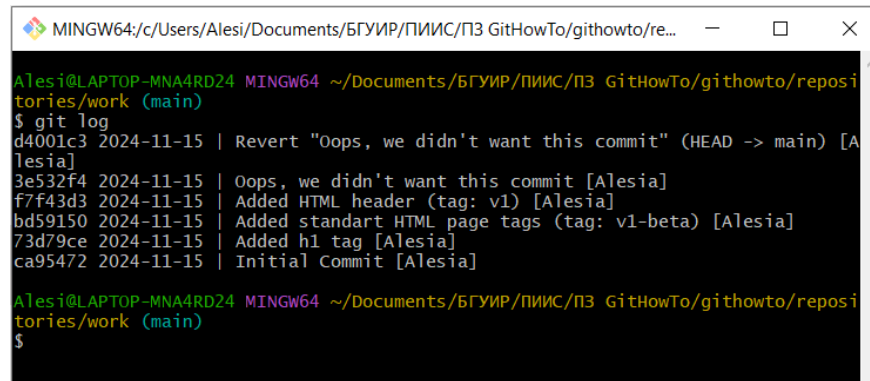
Мы уже видели команду reset и использовали ее для согласования области подготовки с выбранным коммитом (в предыдущем уроке мы использовали коммит HEAD).

Если выполнить команду reset с указанием ссылки на коммит (т.е. метки HEAD, имени ветки или тега, хеша коммита), то команда...

1. Изменит текущую ветку, чтобы она указывала на указанный коммит.
2. Опционально сбросит область подготовки до соответствия с указанным коммитом.
3. Опционально сбросит рабочую директорию до соответствия с указанным коммитом.

Давайте сделаем быструю проверку нашей истории коммитов.

```
git log
```



```
MINGW64: c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git log
d4001c3 2024-11-15 | Revert "Oops, we didn't want this commit" (HEAD -> main) [Alesia]
3e532f4 2024-11-15 | Oops, we didn't want this commit [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$
```

Рисунок 48 – Проверка истории коммитов

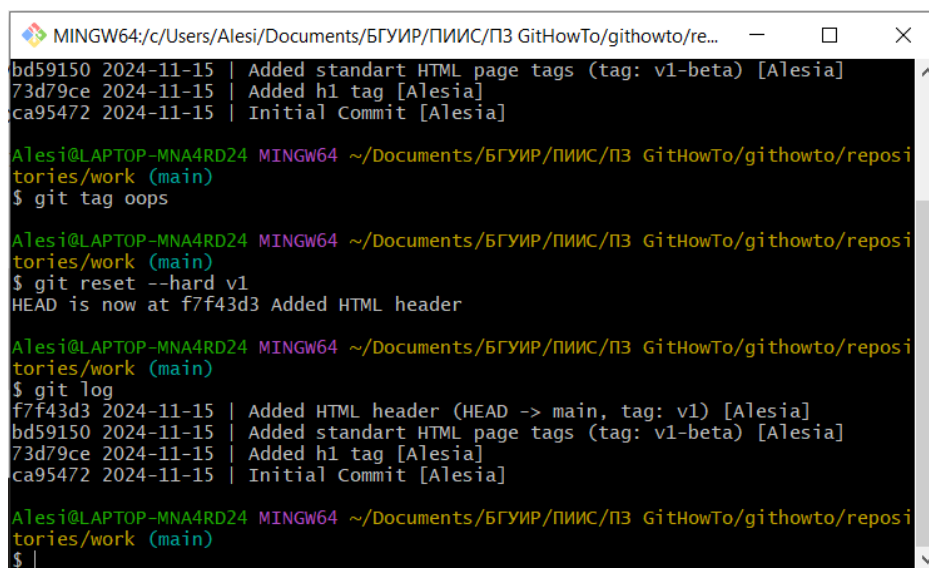
Мы видим, что два последних коммита в этой ветке - «Oops» и «Revert Oops». Давайте удалим их с помощью сброса.

Но прежде чем удалить коммиты, давайте отметим последний коммит тегом, чтобы потом можно было его найти.

```
git tag oops
```

Глядя на историю лога (см. выше), мы видим, что коммит с тегом v1 является коммитом, предшествующим ошибочному коммиту. Давайте сбросим ветку до этой точки. Поскольку ветка имеет тег, мы можем использовать имя тега в команде сброса reset (если она не имеет тега, мы можем использовать хеш коммита).

```
git reset --hard v1
git log
```



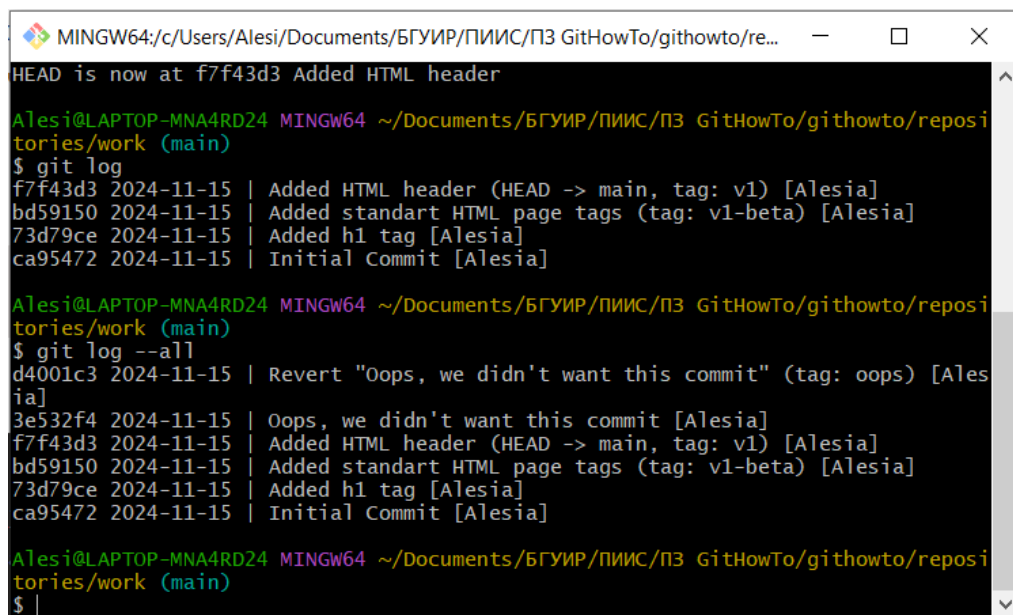
```
MINGW64: c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git tag oops
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git reset --hard v1
HEAD is now at f7f43d3 Added HTML header
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git log
f7f43d3 2024-11-15 | Added HTML header (HEAD -> main, tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$
```

Рисунок 49 – Сброс к предыдущему коммиту

Наша ветка main теперь указывает на коммит v1, а коммитов "Revert Oops" и "Oops" в ветке уже нет. Параметр --hard указывает, что рабочая директория должна быть приведена к тому состоянию, которое соответствует HEAD-коммиту ветки.

Что же случается с ошибочными коммитами? Оказывается, что коммиты все еще находятся в репозитории. На самом деле, мы все еще можем на них ссылаться. Помните, в начале этого урока мы создали для отмененного коммита тег oops? Давайте посмотрим на все коммиты.

```
git log --all
```

A screenshot of a terminal window titled 'MINGW64: c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...'. The terminal shows the output of 'git log' and 'git log --all'. The first 'git log' shows the current history from HEAD (f7f43d3) to the initial commit (ca95472). The second 'git log --all' shows the full history, including the 'Oops' commit (3e532f4) and its revert (d4001c3).

```
MINGW64: c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
HEAD is now at f7f43d3 Added HTML header
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log
f7f43d3 2024-11-15 | Added HTML header (HEAD -> main, tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log --all
d4001c3 2024-11-15 | Revert "Oops, we didn't want this commit" (tag: oops) [Ales
ia]
3e532f4 2024-11-15 | Oops, we didn't want this commit [Alesia]
f7f43d3 2024-11-15 | Added HTML header (HEAD -> main, tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 50 – Просмотр всех коммитов

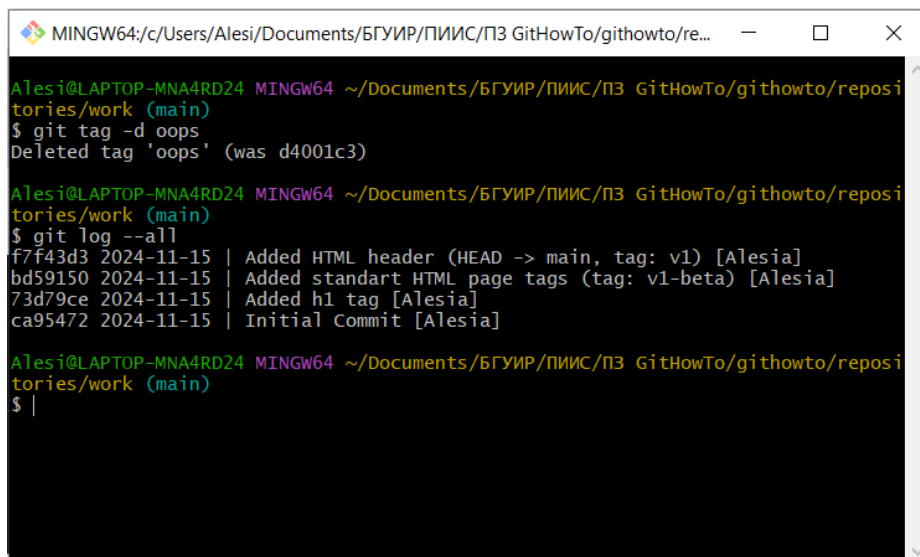
Сброс в локальных ветках, как правило, безопасен. Последствия любой «аварии» как правило, можно восстановить простым сбросом с помощью нужного коммита.

Однако, если ветка уже стала общедоступной на удаленных репозиториях, сброс может сбить с толку других пользователей ветки.

## Задание 16. Удаление тега oops

Тег oops свою функцию выполнил, давайте удалим его. Это позволит внутреннему механизму Git убрать остаточные коммиты, на которые теперь не ссылаются никакие ветки или теги.

```
git tag -d oops
git log --all
```



```
MINGW64:/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ git tag -d oops
Deleted tag 'oops' (was d4001c3)

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
$ git log --all
f7f43d3 2024-11-15 | Added HTML header (HEAD -> main, tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]
```

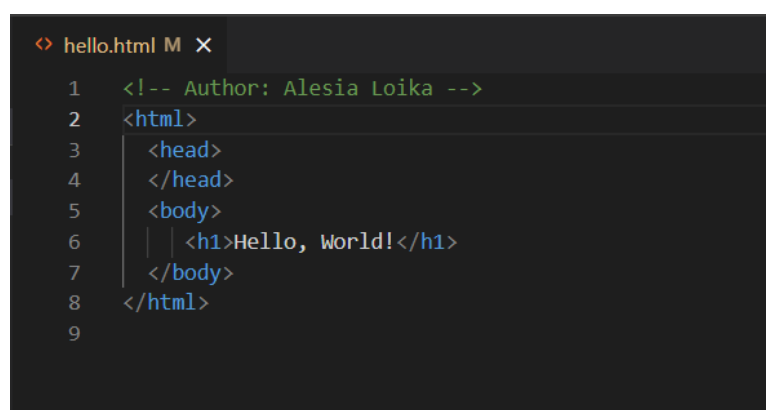
Рисунок 51 – Удаление тега oops

Тег oops больше не будет отображаться в репозитории.

## Задание 17. Внесение изменений в коммиты

Добавьте в страницу комментариев автора.

```
<!-- Author: Alexander Shvets -->
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

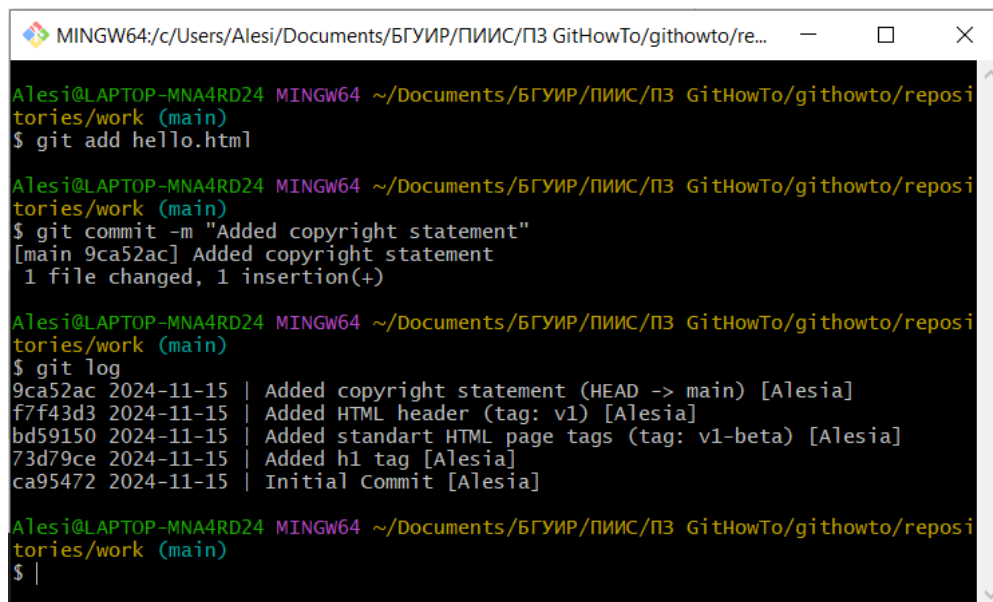


```
hello.html M x
1 <!-- Author: Alesia Loika -->
2 <html>
3   <head>
4   </head>
5   <body>
6   | <h1>Hello, World!</h1>
7   </body>
8 </html>
9
```

Рисунок 52 – Изменение файла helo.html

Затем выполните:

```
git add hello.html
git commit -m "Added copyright statement"
git log
```



```
MINGW64/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ git add hello.html

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ git commit -m "Added copyright statement"
[main 9ca52ac] Added copyright statement
1 file changed, 1 insertion(+)

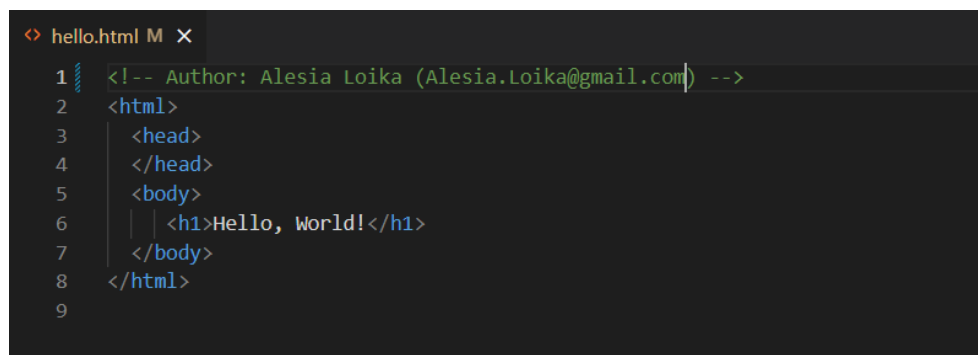
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ git log
9ca52ac 2024-11-15 | Added copyright statement (HEAD -> main) [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repo...
$ |
```

Рисунок 53 – Создание коммита

Однако после создания коммита вы понимаете, что любой хороший комментарий должен включать электронную почту автора. Обновите страницу `hello.html`, включив в нее email.

```
<!-- Author: Alexander Shvets (alex@githowto.com) -->
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```



```
hello.html M X
1 <!-- Author: Alesia Loika (Alesia.Loika@gmail.com) -->
2 <html>
3   <head>
4   </head>
5   <body>
6     <h1>Hello, World!</h1>
7   </body>
8 </html>
9
```

Рисунок 54 – Изменение файла `hello.html`

Мы действительно не хотим создавать отдельный коммит только ради электронной почты. Давайте изменим предыдущий коммит, включив в него адрес электронной почты.

```
git add hello.html
git commit --amend -m "Added copyright statement with email"
```

```
MINGW64:/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log
9ca52ac 2024-11-15 | Added copyright statement (HEAD -> main) [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git add hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git commit --amend -m "Added copyright statement with email"
[main b8322d3] Added copyright statement with email
Date: Fri Nov 15 22:45:31 2024 +0300
1 file changed, 1 insertion(+)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 55 – Изменение коммита

Просмотр истории:

`git log`

```
MINGW64:/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git add hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git commit --amend -m "Added copyright statement with email"
[main b8322d3] Added copyright statement with email
Date: Fri Nov 15 22:45:31 2024 +0300
1 file changed, 1 insertion(+)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log
b8322d3 2024-11-15 | Added copyright statement with email (HEAD -> main) [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 56 – Просмотр истории

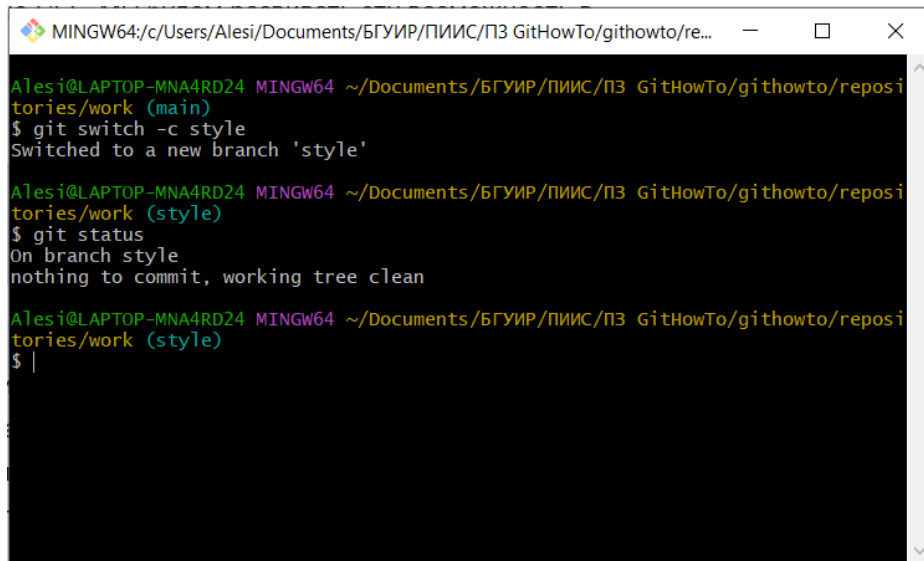
Мы можем увидеть, что оригинальный коммит «автор» заменен коммитом «автор/email». Этого же эффекта можно достичь путем сброса последнего коммита в ветке, и повторного коммита новых изменений.

## Задание 18. Создание ветки

Пришло время сделать нашу страницу более стильной с помощью CSS. Мы будем развивать эту возможность в новой ветке под названием style.



```
git switch -c style
git status
```



```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git switch -c style
Switched to a new branch 'style'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ |
```

Рисунок 57 – Создание новой ветки style

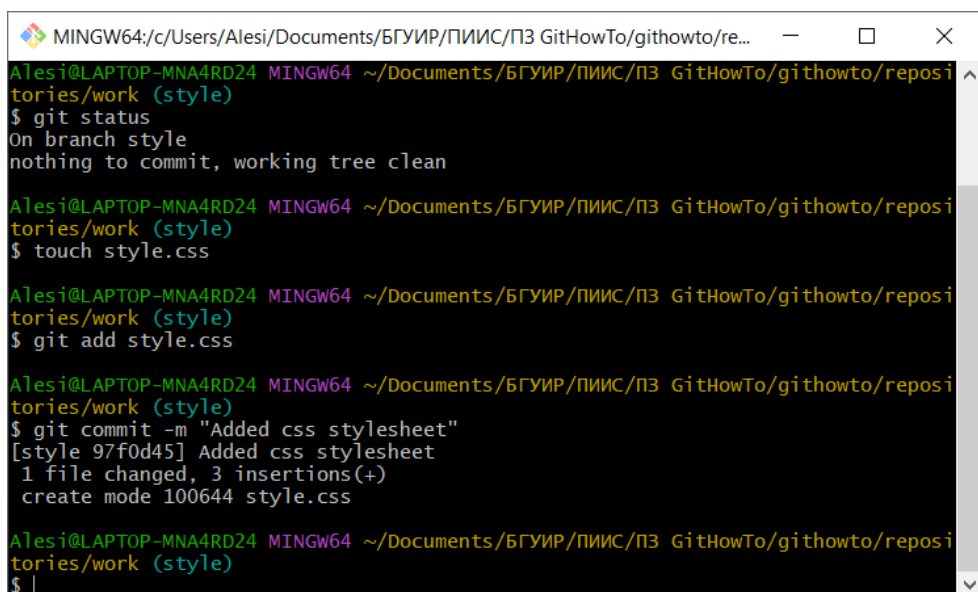
Обратите внимание, что команда `git status` сообщает о том, что вы находитесь в ветке `style`.

Добавьте файл стилей `style.css`

```
touch style.css

h1 {
  color: red;
}

git add style.css
git commit -m "Added css stylesheet"
```



```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ touch style.css

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git add style.css

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git commit -m "Added css stylesheet"
[style 97f0d45] Added css stylesheet
1 file changed, 3 insertions(+)
create mode 100644 style.css

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ |
```

Рисунок 58 – Создание и коммит файла `style.css`

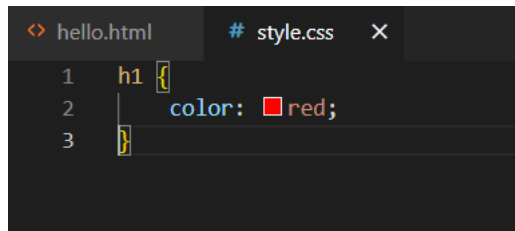


Рисунок 59 – Содержание файла style.css

Измените hello.html, чтобы он использовал style.css.

```
<!-- Author: Alexander Shvets (alex@github.com) -->
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>

git add hello.html
git commit -m "Included stylesheet into hello.html"
```

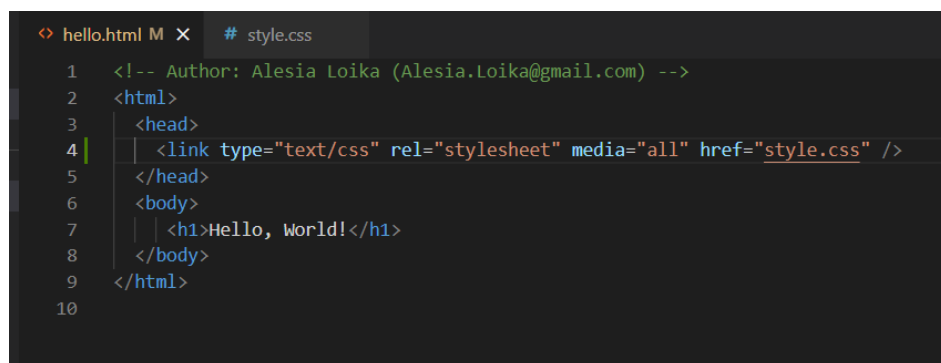


Рисунок 60 – Изменение файла hello.html

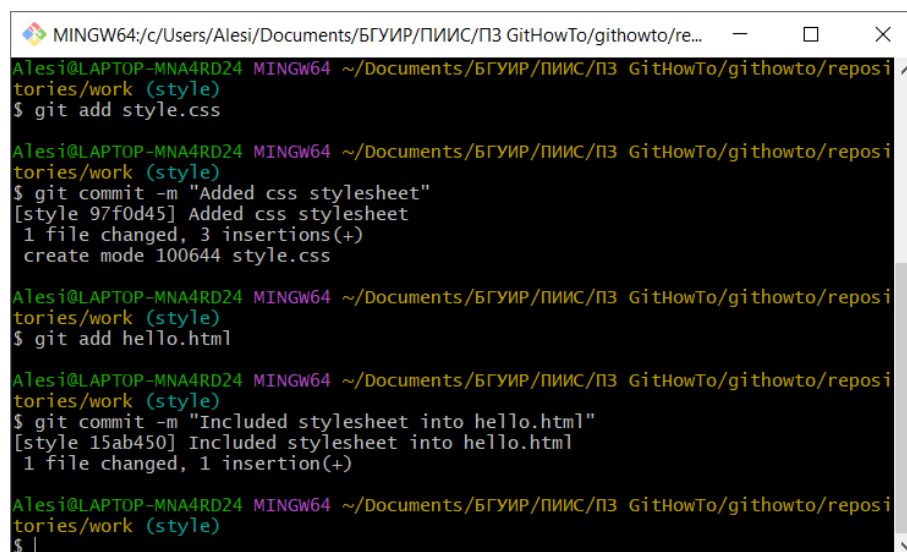


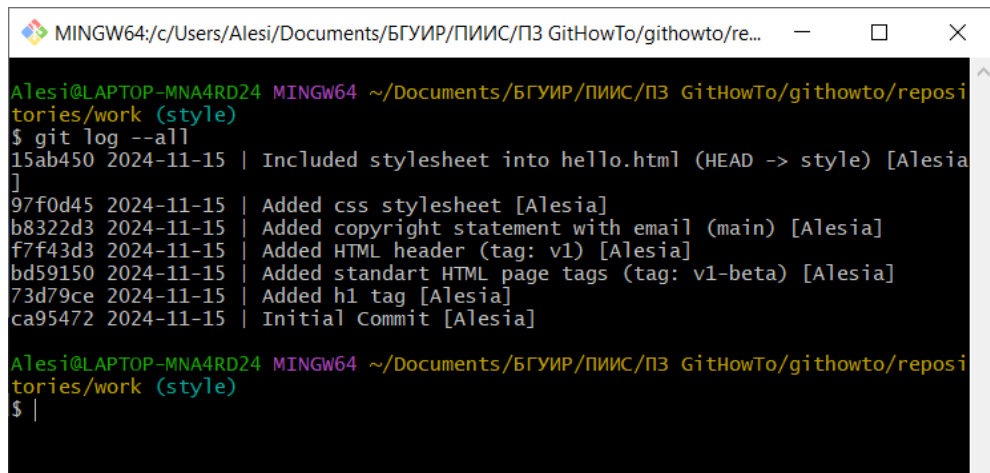
Рисунок 61 – Коммит изменений

Теперь у нас есть новая ветка под названием `style` с двумя новыми коммитами. Далее мы узнаем, как переключаться между ветками.

## Задание 19. Переключение веток

Теперь в вашем проекте есть две ветки:

```
git log --all
```



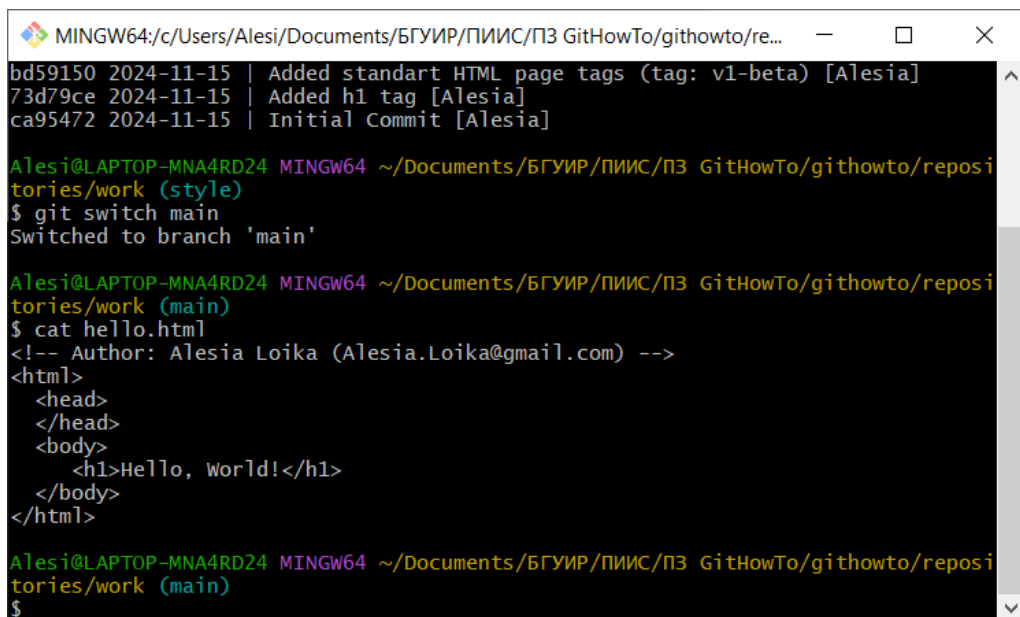
```
MINGW64/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git log --all
15ab450 2024-11-15 | Included stylesheet into hello.html (HEAD -> style) [Alesia]
97f0d45 2024-11-15 | Added css stylesheet [Alesia]
b8322d3 2024-11-15 | Added copyright statement with email (main) [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ |
```

Рисунок 62 – Просмотр веток

Просто используйте команду `git switch` для переключения между ветками.

```
git switch main
cat hello.html
```



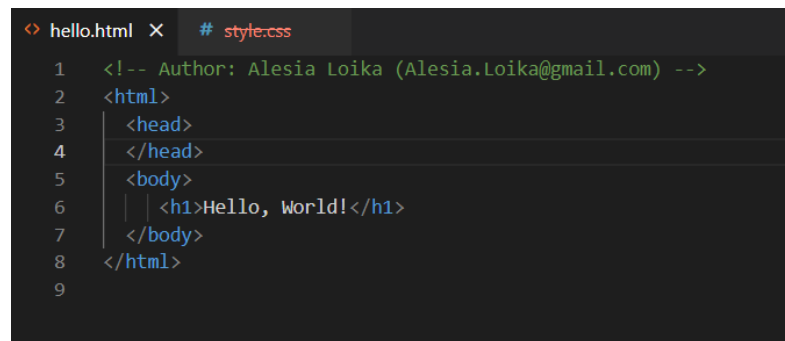
```
MINGW64/c/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git switch main
Switched to branch 'main'

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ cat hello.html
<!-- Author: Alesia Loika (Alesia.Loika@gmail.com) -->
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 63 – Переключение между ветками

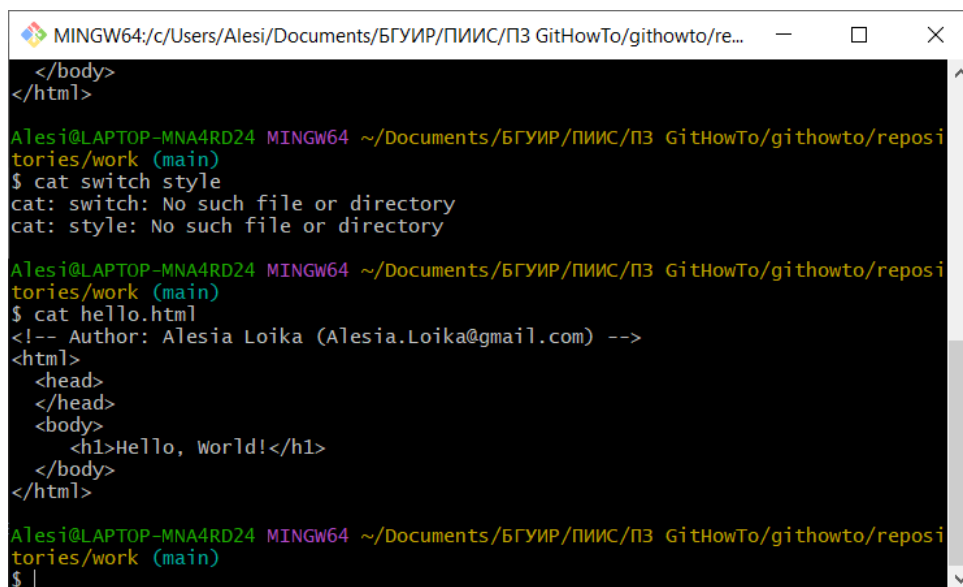


```
hello.html X # style.css
1 <!-- Author: Alesia Loika (Alesia.Loika@gmail.com) -->
2 <html>
3   <head>
4   </head>
5   <body>
6     <h1>Hello, World!</h1>
7   </body>
8 </html>
9
```

Рисунок 64 – Ветка main

Теперь мы находимся в ветке main. Как видите, в hello.html нет никаких следов style.css. Не волнуйтесь, эти изменения все еще есть в репозитории, но мы не можем увидеть их из ветки main. Вернемся к ветке style:

```
git switch style
cat hello.html
```



```
MINGW64/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
</body>
</html>

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ cat switch style
cat: switch: No such file or directory
cat: style: No such file or directory

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ cat hello.html
<!-- Author: Alesia Loika (Alesia.Loika@gmail.com) -->
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 65 – Возвращение к ветке style

Мы вернулись к ветке `style`. Как видите, наши изменения, связанные с CSS, присутствуют.

## Задание 20. Перемещение файлов

Git позволяет просматривать историю изменений конкретного файла. Давайте посмотрим историю изменений файла hello.html перед его переименованием.

```
git log hello.html
git log style.css
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log hello.html
b8322d3 2024-11-15 | Added copyright statement with email (HEAD -> main) [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log style.css
fatal: ambiguous argument 'style.css': unknown revision or path not in the worki
ng tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...] -- [<file>...]'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 66 – Просмотр истории изменений в файлах hello.html и style.css

Команда show используется для просмотра изменений в конкретном коммите. Посмотрим изменения в файле hello.html в коммите, с тегом v1 (можно использовать любую ссылку на коммит, например, метку HEAD, хеш коммита, имя ветки или тега и т.д.).

```
git show v1
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
ng tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...] -- [<file>...]'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git show v1
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]

diff --git a/hello.html b/hello.html
index 9b5ef91..6c140e2 100644
--- a/hello.html
+++ b/hello.html
@@ -1,4 +1,6 @@
<html>
+ <head>
+ </head>
<body>
  <h1>Hello, World!</h1>
</body>

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 67 – Просмотр изменений в коммите

Давайте переименуем наш файл hello.html в index.html с помощью стандартной команды mv и посмотрим, что из этого получится.

```
mv hello.html index.html
git status
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Switched to branch 'style'
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ mv hello.html index.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   index.html

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    deleted:    hello.html
    modified:   index.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$
```

Рисунок 68 – Переименование файла hello.html в index.html

Git воспринимает наше изменение так, будто файл был удален и создан заново. Это тревожный звоночек. Нам нужно сообщить Git, что мы именно переименовали файл, а не удалили его и сразу создали новый. В простейшем случае Git сам поймёт, что файл был переименован, как только мы добавим его в индекс:

```
git add .
git status
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    deleted:    hello.html
    modified:   index.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git add .

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    renamed:    hello.html -> index.html

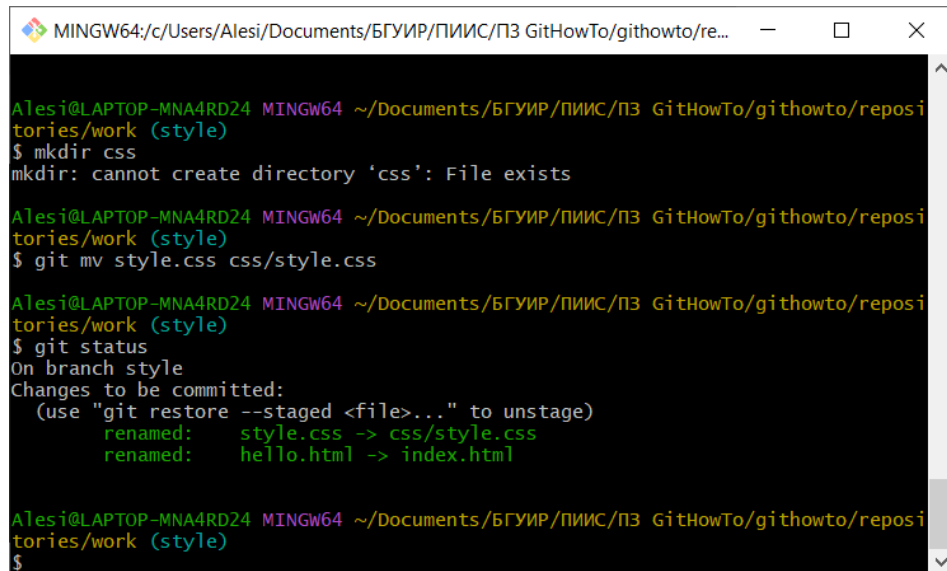
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$
```

Рисунок 69 – Переименование файла в Git

В большинстве операционных систем переименование и перемещение файлов — это одно и то же. Итак, давайте переместим наш файл style.css в директорию css, но на этот раз сделаем это безопасно с помощью команды git

mv. Эта команда гарантирует, что перемещение будет записано в истории Git как перемещение.

```
mkdir css
git mv style.css css/style.css
git status
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ mkdir css
mkdir: cannot create directory 'css': File exists

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git mv style.css css/style.css

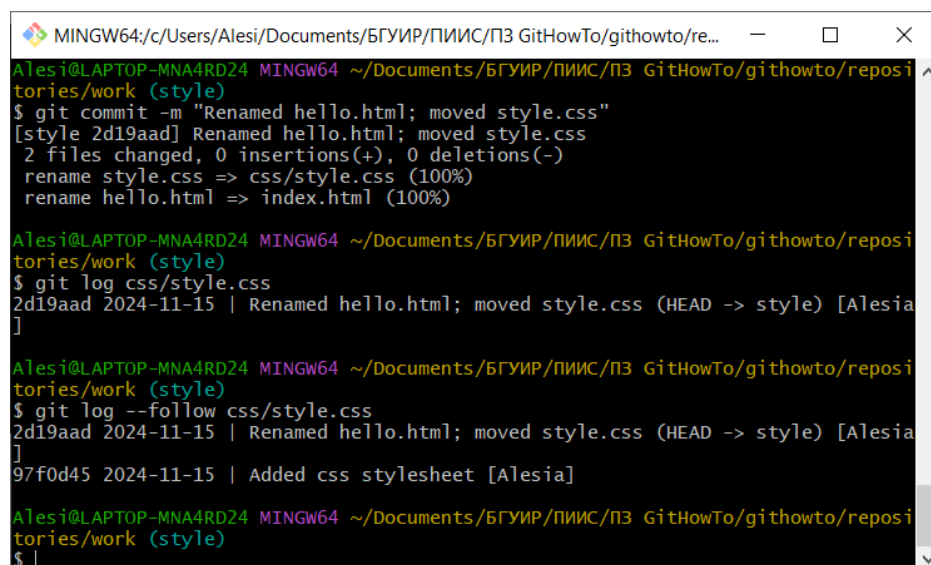
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git status
On branch style
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    style.css -> css/style.css
        renamed:    hello.html -> index.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$
```

Рисунок 70 – Перемещение файла в Git

Давайте закоммитим наши изменения и проверим историю изменений в файле css/styles.css. Для просмотра истории файла до его перемещения нам потребуется добавить опцию --follow. Выполним оба варианта команды, чтобы понять разницу.

```
git commit -m "Renamed hello.html; moved style.css"
git log css/style.css
git log --follow css/style.css
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git commit -m "Renamed hello.html; moved style.css"
[style 2d19aad] Renamed hello.html; moved style.css
2 files changed, 0 insertions(+), 0 deletions(-)
rename style.css => css/style.css (100%)
rename hello.html => index.html (100%)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git log css/style.css
2d19aad 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> style) [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git log --follow css/style.css
2d19aad 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> style) [Alesia]
97f0d45 2024-11-15 | Added css stylesheet [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$
```

Рисунок 71 – Коммит изменений

## Задание 21. Изменения в ветке main

Как я уже говорил, Git позволяет работать с несколькими ветками одновременно. Это очень удобно при работе в команде, поскольку люди могут параллельно работать над разными функциями. Это также полезно при работе в одиночку: разрабатывая функции в отдельных ветках, вы можете исправлять ошибки и выпускать небольшие обновления, используя стабильный код в ветке main.

Создадим файл README. В нем будет рассказано о сути нашего проекта.

This is the Hello World example from the GitHowTo tutorial.

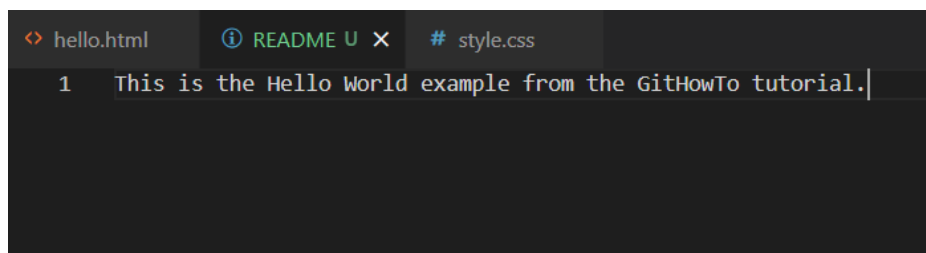


Рисунок 72 – Созданный файл README

Сделайте коммит файла README в ветку main. В настоящее время мы находимся в ветке style. Файл README не является частью этой ветки, поэтому перед коммитом мы должны переключиться на ветку main.

```
git switch main
git add README
git commit -m "Added README"
```

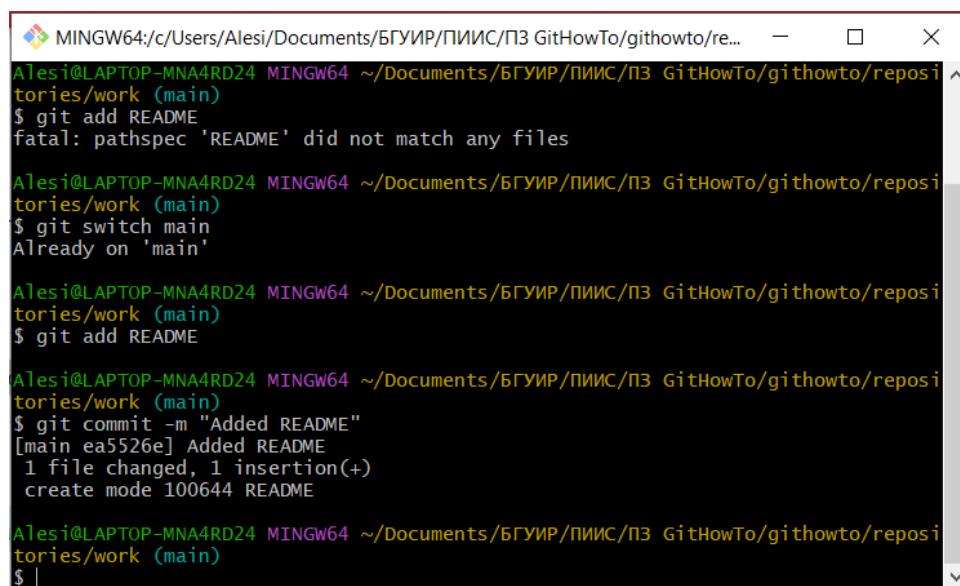


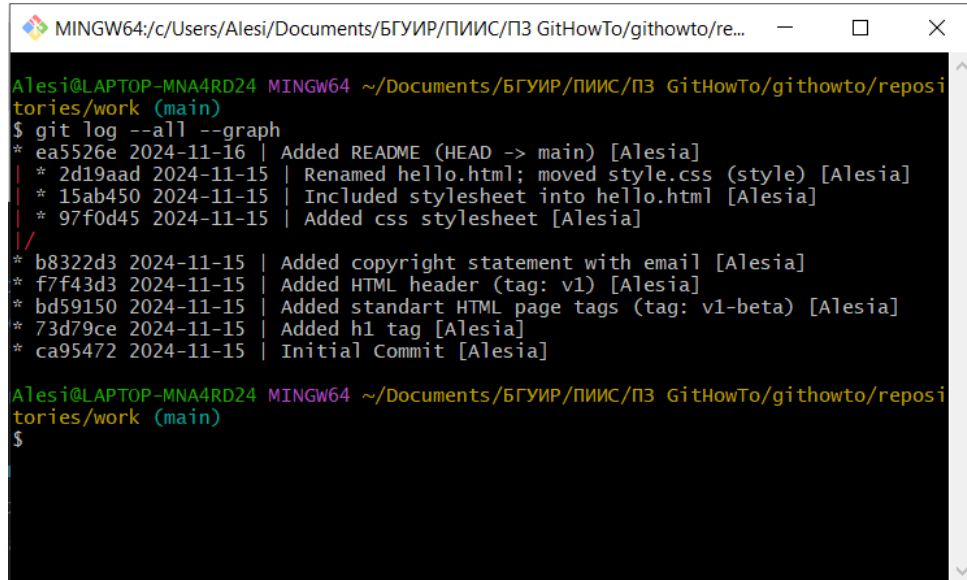
Рисунок 73 – Коммит файла README



## Задание 22. Просмотр отличающихся веток

Теперь у нас есть две расходящиеся ветки в репозитории. Используйте следующую команду log для просмотра веток и их расхождения.

```
git log --all --graph
```



```
MINGW64:/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ git log --all --graph
* ea5526e 2024-11-16 | Added README (HEAD -> main) [Alesia]
| * 2d19aad 2024-11-15 | Renamed hello.html; moved style.css (style) [Alesia]
| * 15ab450 2024-11-15 | Included stylesheet into hello.html [Alesia]
| * 97f0d45 2024-11-15 | Added css stylesheet [Alesia]
|/
* b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
* f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
* bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
* 73d79ce 2024-11-15 | Added h1 tag [Alesia]
* ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$
```

Рисунок 74 – Просмотр текущих веток

Опция --all гарантирует, что мы видим все ветки, так как по умолчанию в логе показывается только текущая ветка.

Опция --graph добавляет простое дерево коммитов, представленное в виде простых текстовых линий. Мы видим обе ветки (style и main) причём ветка main помечена как HEAD, что означает, что она является текущей. Общим предком для обеих веток является ветка, в которую был внесен коммит «Added copyright statement with email».

## Задание 23. Слияние

Слияние переносит изменения из двух веток в одну. Давайте вернемся к ветке style и сольем main со style.

```
git switch style
git merge main
git log --all --graph
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git switch style
Switched to branch 'style'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git merge main
Merge made by the 'ort' strategy.
 README | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 README

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ git log --all --graph
* 1e69227 2024-11-16 | Merge branch 'main' into style (HEAD -> style) [Alesia]
|
| * ea5526e 2024-11-16 | Added README (main) [Alesia]
| * 2d19aad 2024-11-15 | Renamed hello.html; moved style.css [Alesia]
| * 15ab450 2024-11-15 | Included stylesheet into hello.html [Alesia]
| * 97f0d45 2024-11-15 | Added css stylesheet [Alesia]
|
| * b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
| * f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
| * bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
| * 73d79ce 2024-11-15 | Added h1 tag [Alesia]
| * ca95472 2024-11-15 | Initial Commit [Alesia]
|
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (style)
$ |
```

Рисунок 75 – Слияние веток

Путем периодического слияния ветки main с веткой style вы можете переносить из main любые изменения и поддерживать совместимость изменений style с изменениями в основной ветке.

Но что если изменения в ветке main конфликтуют с изменениями в style?

## Задание 24. Создание конфликта

При слиянии двух веток Git пытается перенести изменения из одной ветки в другую. Если в обеих ветках была изменена одна и та же часть файла, Git может не справиться с автоматическим слиянием изменений. В этом случае Git сообщит о конфликте и попросит разрешить его вручную. В этом уроке мы смоделируем конфликт, а затем научимся его разрешать.

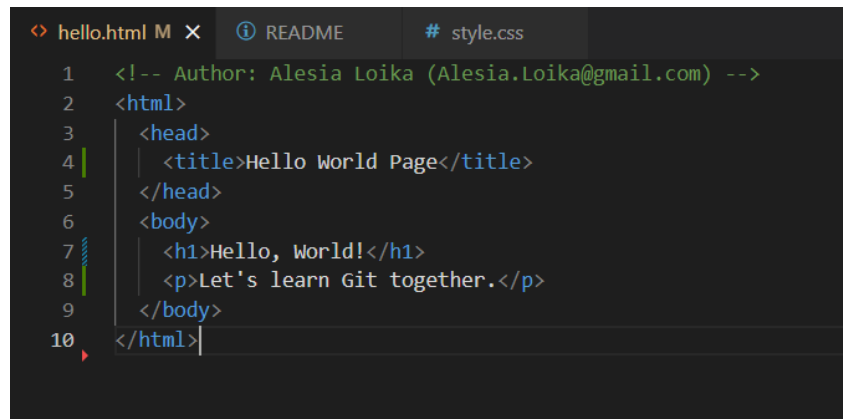
Помните, что в нашей ветке main страница по-прежнему называется hello.html? Переключитесь обратно на ветку main и внесите следующие изменения:

```
git switch main

<!-- Author: Alexander Shvets (alex@ghowto.com) -->
<html>
  <head>
    <title>Hello World Page</title>
  </head>
  <body>
```

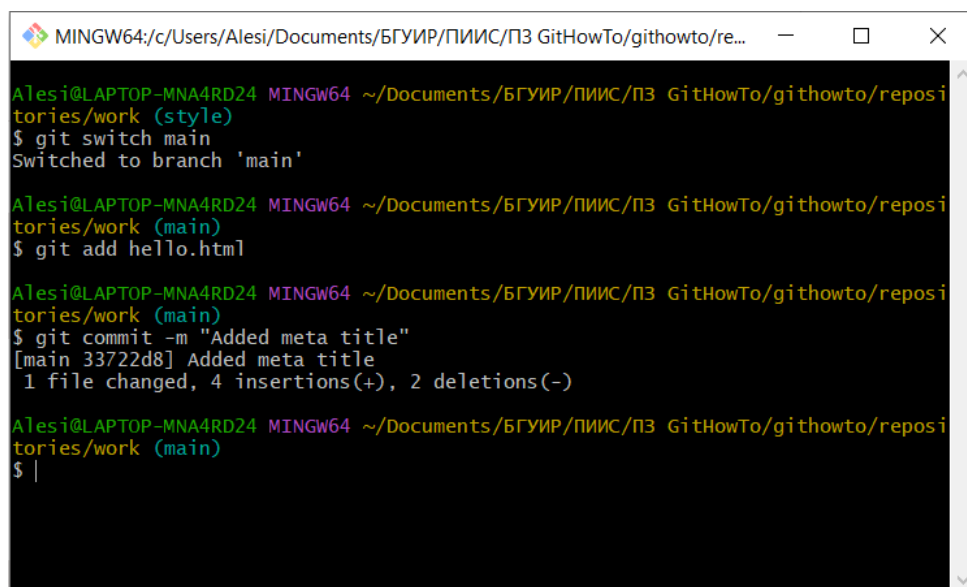
```
<h1>Hello, World!</h1>
<p>Let's learn Git together.</p>
</body>
</html>

git add hello.html
git commit -m "Added meta title"
```

A screenshot of a code editor with a dark theme. The editor has three tabs at the top: 'hello.html M', 'README', and 'style.css'. The 'hello.html' tab is active, showing the following HTML code:

```
1 <!-- Author: Alesia Loika (Alesia.Loika@gmail.com) -->
2 <html>
3   <head>
4     <title>Hello World Page</title>
5   </head>
6   <body>
7     <h1>Hello, World!</h1>
8     <p>Let's learn Git together.</p>
9   </body>
10 </html>
```

Рисунок 76 – Изменение файла hello.html

A screenshot of a terminal window with a black background and green text. The window title is 'MINGW64: c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/re...'. The terminal shows the following commands and output:

```
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (style)
$ git switch main
Switched to branch 'main'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ git add hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ git commit -m "Added meta title"
[main 33722d8] Added meta title
1 file changed, 4 insertions(+), 2 deletions(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
tories/work (main)
$ |
```

Рисунок 77 – Создание коммита

Просмотр веток:

```
git log --all --graph
```



Похоже, что у нас возник конфликт. Ничего удивительного! Посмотрим, что скажет по этому поводу Git:

```
git status
```



```
MINGW64~/c/Users/Alesj/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repo...
Alesj@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repo...
$ git merge main
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.

Alesj@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repo...
$ git status
On branch style
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

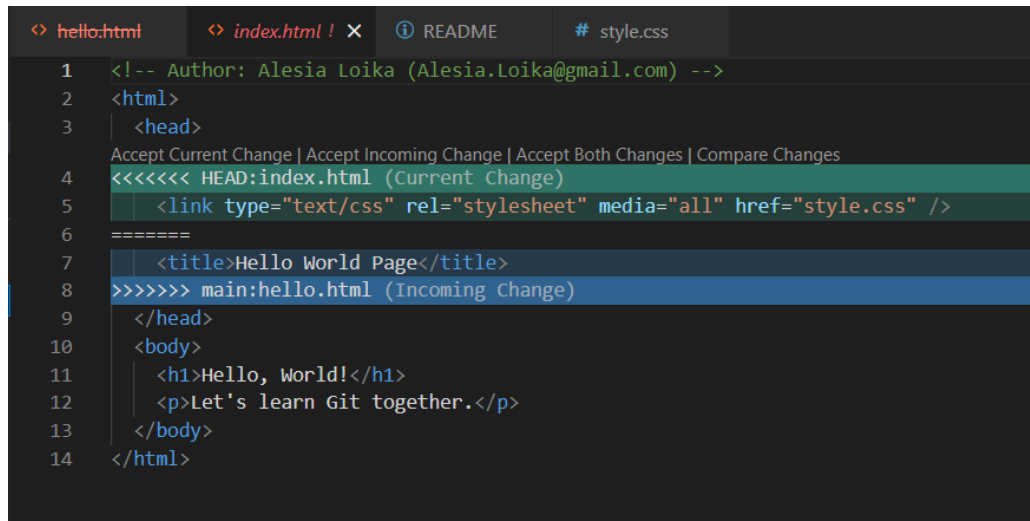
Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   index.html

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

Alesj@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/repo...
$
```

Рисунок 80 – Просмотр состояния Git

Если открыть файл index.html, то можно увидеть:



```
<> hello.html  <> index.html ! x  ⓘ README  # style.css
1  <!-- Author: Alesia Loika (Alesia.Loika@gmail.com) -->
2  <html>
3  <head>
4  Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
5  <<<<<<< HEAD:index.html (Current Change)
6  <link type="text/css" rel="stylesheet" media="all" href="style.css" />
7  =====
8  <title>Hello World Page</title>
9  >>>>>>> main:hello.html (Incoming Change)
10 </head>
11 <body>
12   <h1>Hello, World!</h1>
13   <p>Let's learn Git together.</p>
14 </body>
15 </html>
```

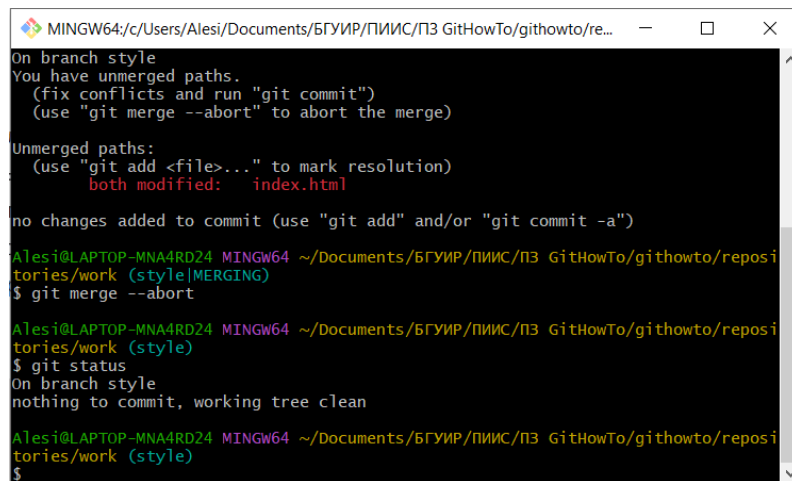
Рисунок 81 – Содержание файла index.html

Часть между <<<<<<< >>>>>>> является конфликтом. Верхняя часть соответствует ветке style, которая является текущей веткой (или HEAD) репозитория. Нижняя часть соответствует изменениям из ветки main. Git не может решить, какие изменения применить, поэтому он просит вас разрешить конфликт вручную. Вы можете оставить изменения из ветки style или из main, либо объединить их любым удобным способом. Вы также можете внести в файл любые другие изменения.

Кстати, вы заметили, что наше второе изменение, тег <p>, не является частью конфликта? Это потому, что Git сумел автоматически объединить ее.

Прежде чем мы приступим к разрешению нашего конфликта, хочу заметить, что сразу бросаться к разрешению конфликта не всегда оптимально. Конфликт может быть вызван изменениями, о которых вы не знаете. Или же изменения слишком велики, чтобы разрешить конфликт сразу. По этой причине Git позволяет прервать слияние и вернуться к предыдущему состоянию. Для этого можно воспользоваться командой `git merge --abort`, как это было предложено командой `status`, которую мы выполнили ранее.

```
git merge --abort
git status
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
On branch style
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   index.html

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

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style|MERGING)
$ git merge --abort

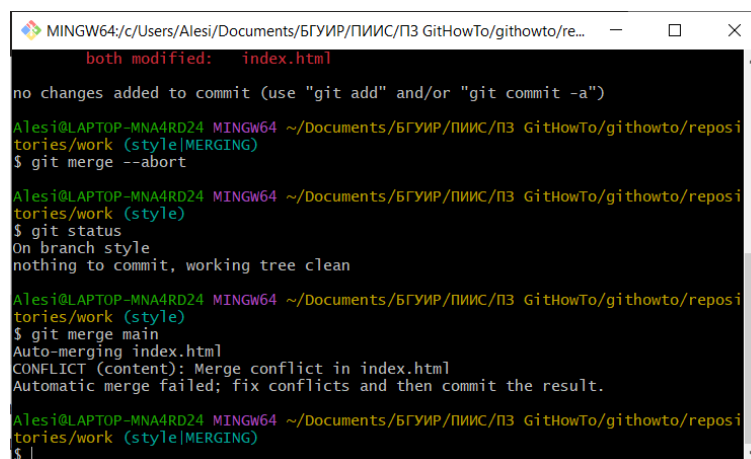
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$
```

Рисунок 82 – Отмена слияния и возврат к предыдущему состоянию

После небольшой медитации мы готовы к разрешению конфликта. Давайте снова запустим объединение.

```
git merge main
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
    both modified:   index.html

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

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style|MERGING)
$ git merge --abort

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
nothing to commit, working tree clean

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git merge main
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style|MERGING)
$
```

Рисунок 83 – Конфликт веток

Чтобы разрешить конфликт, нужно отредактировать файл до состояния, которое нас устраивает, и затем закоммитить его как обычно. В нашем случае мы объединим изменения из обеих веток. Поэтому мы отредактируем файл до следующего состояния:

```
<!-- Author: Alexander Shvets (alex@github.com) -->
<html>
  <head>
    <title>Hello World Page</title>
    <link type="text/css" rel="stylesheet" media="all" href="style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
    <p>Let's learn Git together.</p>
  </body>
</html>
```

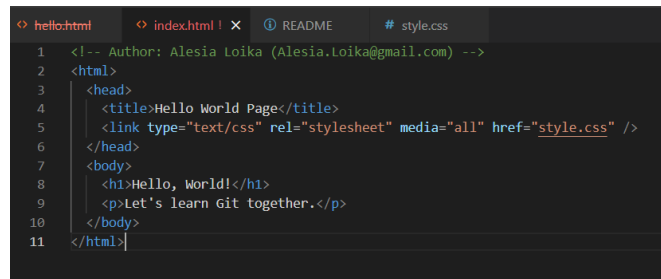


Рисунок 84 – Изменение файла index.html

Закоммитьте разрешенный конфликт:

```
git add index.html
git commit -m "Resolved merge conflict"
git status
git log --all --graph
```

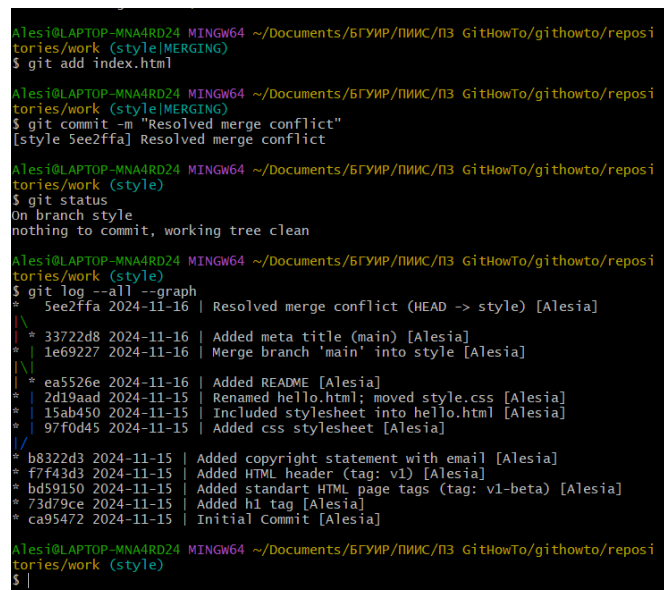
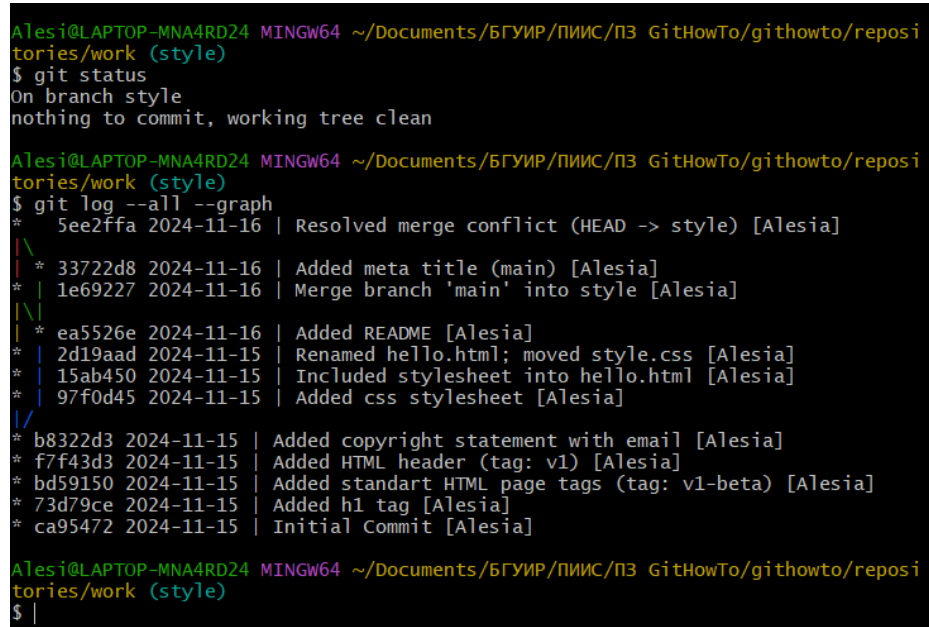


Рисунок 85 – Разрешение конфликта

Давайте посмотрим на текущее состояние нашего хранилища и убедимся, что все в порядке:

```
git status
git log --all --graph
```



```
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
nothing to commit, working tree clean

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git log --all --graph
*   5ee2ffa 2024-11-16 | Resolved merge conflict (HEAD -> style) [Alesia]
|
| * 33722d8 2024-11-16 | Added meta title (main) [Alesia]
| * 1e69227 2024-11-16 | Merge branch 'main' into style [Alesia]
|
| * ea5526e 2024-11-16 | Added README [Alesia]
| * 2d19aad 2024-11-15 | Renamed hello.html; moved style.css [Alesia]
| * 15ab450 2024-11-15 | Included stylesheet into hello.html [Alesia]
| * 97f0d45 2024-11-15 | Added css stylesheet [Alesia]
|/
|
| * b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
| * f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
| * bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
| * 73d79ce 2024-11-15 | Added h1 tag [Alesia]
| * ca95472 2024-11-15 | Initial Commit [Alesia]
|/
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ |
```

Рисунок 86 – Проверка состояния Git

## Задание 26. rebase против merge

Давайте рассмотрим различия между слиянием и перебазированием. Для того чтобы это сделать, нам нужно вернуться в репозиторий в момент до первого слияния, а затем повторить те же действия, но с использованием перебазирования вместо слияния.

Мы будем использовать команду `reset` для возврата веток к предыдущему состоянию.

## Задание 27. Сброс ветки style

Давайте вернемся во времени на ветке `style` к точке перед тем, как мы слили ее с веткой `main`. Мы можем сбросить ветку к любому коммиту при помощи команды `reset`. По сути, это изменение указателя ветки на любую точку дерева коммитов.

В этом случае мы хотим вернуться в ветке `style` в точку перед слиянием с `main`. Нам необходимо найти последний коммит перед слиянием.

```
git switch style
git log --graph
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git switch style
Already on 'style'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git log --graph
* 5ee2ffa 2024-11-16 | Resolved merge conflict (HEAD -> style) [Alesia]
|
| * 33722d8 2024-11-16 | Added meta title (main) [Alesia]
| * 1e69227 2024-11-16 | Merge branch 'main' into style [Alesia]
|
| * ea5526e 2024-11-16 | Added README [Alesia]
| * 2d19aad 2024-11-15 | Renamed hello.html; moved style.css [Alesia]
| * 15ab450 2024-11-15 | Included stylesheet into hello.html [Alesia]
| * 97f0d45 2024-11-15 | Added css stylesheet [Alesia]
|/
* b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
* f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
* bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
* 73d79ce 2024-11-15 | Added h1 tag [Alesia]
* ca95472 2024-11-15 | Initial Commit [Alesia]
```

Рисунок 87 – Возврат к ветке style

Это немного трудно читать, но, глядя на данные, мы видим, что коммит «Renamed hello.html; moved style.css» был последним на ветке style перед слиянием. Давайте сбросим ветку style к этому коммиту. Чтобы сослаться на этот коммит, мы либо используем его хеш, либо посчитаем, что этот коммит находится за 2 коммита до HEAD, то есть HEAD~2 в нотации Git.

```
git reset --hard HEAD~2
```

Теперь проверим историю ветки style. В истории не должно быть коммитов слияния.

```
git log --graph
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
* 73d79ce 2024-11-15 | Added h1 tag [Alesia]
* ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git reset --hard HEAD~2
HEAD is now at 2d19aad Renamed hello.html; moved style.css

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git log --graph
* 2d19aad 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> style) [Alesia]
|
| * 15ab450 2024-11-15 | Included stylesheet into hello.html [Alesia]
| * 97f0d45 2024-11-15 | Added css stylesheet [Alesia]
| * b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
| * f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
| * bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
| * 73d79ce 2024-11-15 | Added h1 tag [Alesia]
| * ca95472 2024-11-15 | Initial Commit [Alesia]
|/

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$
```

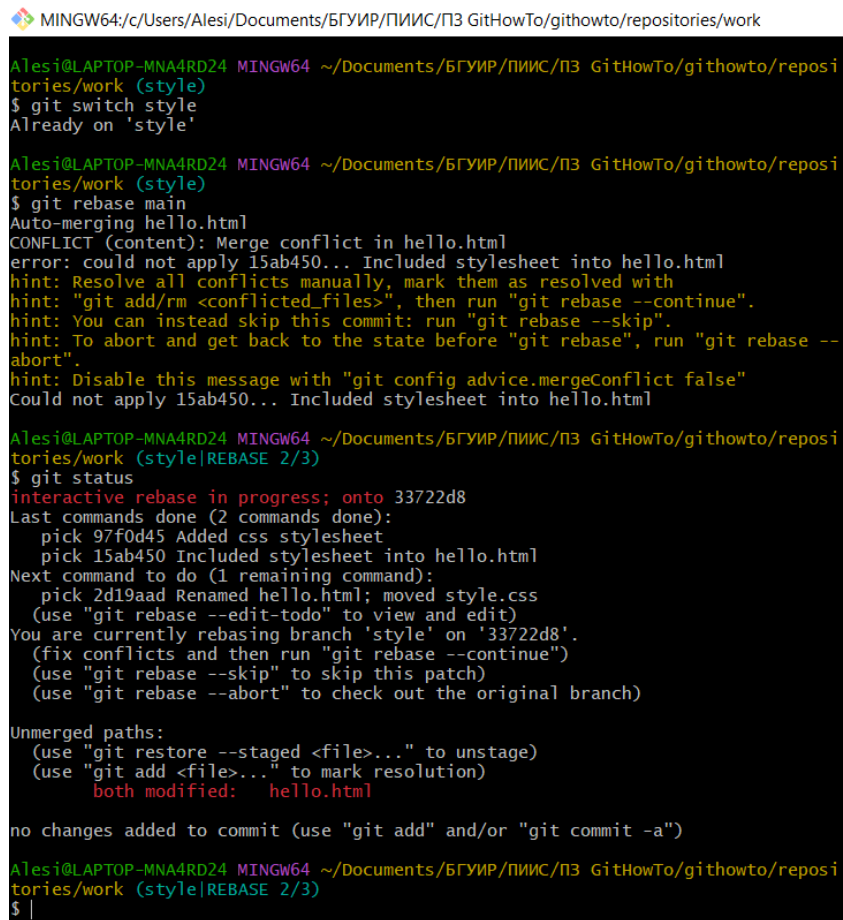
Рисунок 88 – Проверка ветки style

## Задание 28. Перебазирование

Мы вернули ветку style к точке перед первым слиянием. При этом в ветке main есть два коммита, которых нет в ветке style: новый файл README и конфликтующее изменение в файле index.html. На этот раз мы перенесем эти изменения в ветку style с помощью команды rebase, а не merge.

Перебазируем ветку style на ветку main:

```
git switch style
git rebase main
git status
```



```
MINGW64/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work (style)
$ git switch style
Already on 'style'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work (style)
$ git rebase main
Auto-merging hello.html
CONFLICT (content): Merge conflict in hello.html
error: could not apply 15ab450... Included stylesheet into hello.html
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply 15ab450... Included stylesheet into hello.html

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work (style|REBASE 2/3)
$ git status
interactive rebase in progress; onto 33722d8
Last commands done (2 commands done):
  pick 97f0d45 Added css stylesheet
  pick 15ab450 Included stylesheet into hello.html
Next command to do (1 remaining command):
  pick 2d19aad Renamed hello.html; moved style.css
(use "git rebase --edit-todo" to view and edit)
You are currently rebasing branch 'style' on '33722d8'.
(fix conflicts and then run "git rebase --continue")
(use "git rebase --skip" to skip this patch)
(use "git rebase --abort" to check out the original branch)

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   hello.html

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

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work (style|REBASE 2/3)
$ |
```

Рисунок 89 – Перезабазирование ветки style на ветку main

Опять возник конфликт! Обратите внимание, что конфликт возник в файле hello.html, а не в файле index.html, как в прошлый раз. Это связано с тем, что rebase находился в процессе применения изменений style поверх ветки main. Файл hello.html в main еще не был переименован, поэтому он все еще имеет старое имя.

При слиянии возник бы «обратный» конфликт. При слиянии изменения ветки main были бы применены поверх ветки style. В ветке style файл переименован, поэтому конфликт возник бы в файле index.html.

Рисунок 90 – Конфликт в файле hello.html

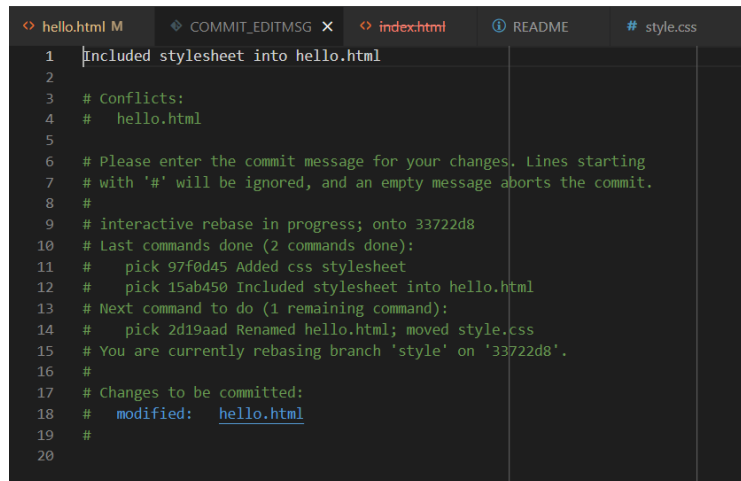
Сам конфликт может быть разрешен тем же способом, что и предыдущий. Сначала мы изменим файл hello.html, чтобы он соответствовал нашим ожиданиям.

```
<!-- Author: Alexander Shvets (alex@github.com) -->
<html>
  <head>
    <title>Hello World Page</title>
    <link type="text/css" rel="stylesheet" media="all" href="style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
    <p>Let's learn Git together.</p>
  </body>
</html>
```

Рисунок 91 – Изменение файла hello.html

Но после этого нам не нужно коммитить изменения. Мы можем просто добавить файл в индекс и продолжить процесс rebase. Вот почему я люблю rebase! Он позволяет мне устранять конфликты, не создавая кучу уродливых конфликтов слияния.

```
git add .
git rebase --continue
```

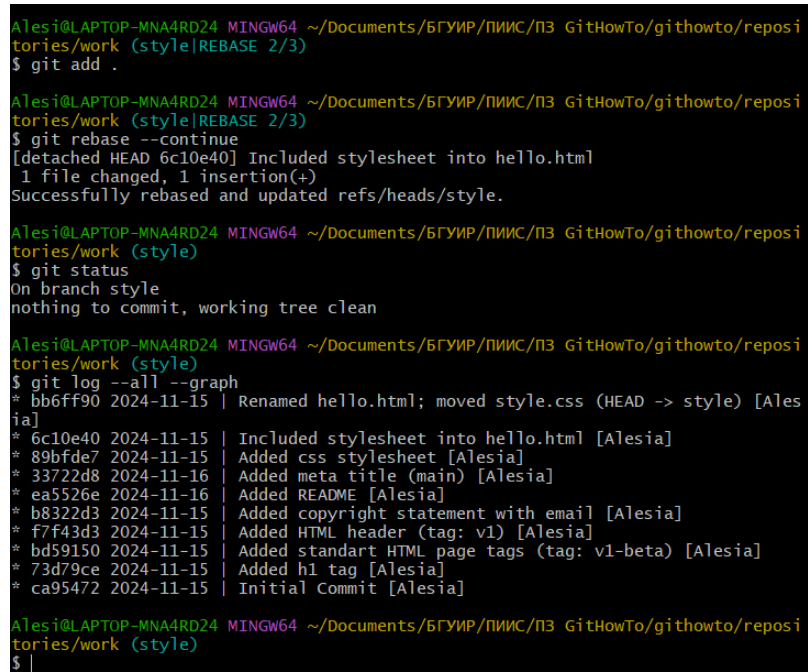


```
1 Included stylesheet into hello.html
2
3 # Conflicts:
4 #   hello.html
5
6 # Please enter the commit message for your changes. Lines starting
7 # with '#' will be ignored, and an empty message aborts the commit.
8 #
9 # interactive rebase in progress; onto 33722d8
10 # Last commands done (2 commands done):
11 #   pick 97f0d45 Added css stylesheet
12 #   pick 15ab450 Included stylesheet into hello.html
13 # Next command to do (1 remaining command):
14 #   pick 2d19aad Renamed hello.html; moved style.css
15 # You are currently rebasing branch 'style' on '33722d8'.
16 #
17 # Changes to be committed:
18 #   modified:   hello.html
19 #
20
```

Рисунок 92 – Открытие редактора

Здесь, скорее всего, Git снова откроет редактор, чтобы позволить нам изменить текст коммита. Мы можем оставить текст без изменений. После сохранения изменений Git завершит процесс rebase, и мы сможем выполнить следующие команды:

```
git status
git log --all --graph
```



```
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style|REBASE 2/3)
$ git add .

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style|REBASE 2/3)
$ git rebase --continue
[detached HEAD 6c10e40] Included stylesheet into hello.html
1 file changed, 1 insertion(+)
Successfully rebased and updated refs/heads/style.

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git status
On branch style
nothing to commit, working tree clean

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git log --all --graph
* bb6ff90 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> style) [Ales
ia]
* 6c10e40 2024-11-15 | Included stylesheet into hello.html [Alesia]
* 89bfde7 2024-11-15 | Added css stylesheet [Alesia]
* 33722d8 2024-11-16 | Added meta title (main) [Alesia]
* ea5526e 2024-11-16 | Added README [Alesia]
* b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
* f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
* bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
* 73d79ce 2024-11-15 | Added h1 tag [Alesia]
* ca95472 2024-11-15 | Initial Commit [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ |
```

Рисунок 92 – Заверение rebase и просмотр состояния Git

Конечный результат перебазирувания очень похож на результат слияния. Ветка style в настоящее время содержит все свои изменения, а также все изменения ветки main. Однако, дерево коммитов значительно отличается. Дерево коммитов ветки style было переписано таким образом, что ветка main

является частью истории коммитов. Это делает цепь коммитов линейной и гораздо более читабельной.

Используйте команду rebase:

- Когда вы получаете изменения из удаленного репозитория и хотите применить их к своей локальной ветке.

- Если вы хотите, чтобы история коммитов была линейной и легко читаемой.

Не используйте команду rebase:

- Если текущая ветка является публичной и общей. Переписывание таких веток будет мешать работе других членов команды.


- Если важна точная история ветки коммитов (поскольку команда rebase переписывает историю коммитов).

Учитывая приведенные выше рекомендации, я предпочитаю использовать команду rebase для краткосрочных, локальных веток и команду merge для веток в публичном репозитории.

## Задание 29. Слияние в ветку main

Мы поддерживали соответствие ветки style с веткой main (с помощью rebase), теперь давайте сольем изменения style в ветку main.

```
git switch main
git merge style
```



```
MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (style)
$ git switch main
Switched to branch 'main'

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git merge style
Updating 33722d8..bb6ff90
Fast-forward
 css/style.css      | 3 +++
 hello.html => index.html | 1 +
 2 files changed, 4 insertions(+)
 create mode 100644 css/style.css
 rename hello.html => index.html (73%)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

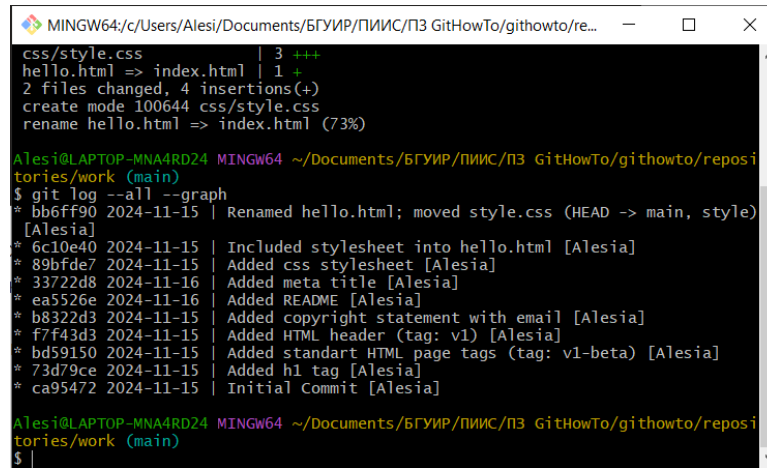
Рисунок 93 – Слияние ветки style в main

Поскольку последний коммит в main предшествует последнему коммиту ветки style, Git может выполнить ускоренное слияние, просто переместив указатель ветки вперед, на тот же коммит, что и ветка style.

При ускоренном слиянии конфликты не возникают. Кроме того, при ускоренном слиянии не создается фиксация слияния.

Просмотрите логи:

```
git log --all --graph
```



```
css/style.css | 3 +++
hello.html => index.html | 1 +
2 files changed, 4 insertions(+)
create mode 100644 css/style.css
rename hello.html => index.html (73%)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git log --all --graph
* bb6ff90 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> main, style)
[Alesia]
* 6c10e40 2024-11-15 | Included stylesheet into hello.html [Alesia]
* 89bfde7 2024-11-15 | Added css stylesheet [Alesia]
* 33722d8 2024-11-16 | Added meta title [Alesia]
* ea5526e 2024-11-16 | Added README [Alesia]
* b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
* f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
* bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
* 73d79ce 2024-11-15 | Added h1 tag [Alesia]
* ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 94 – Просмотр логов

Теперь ветки style и main идентичны.

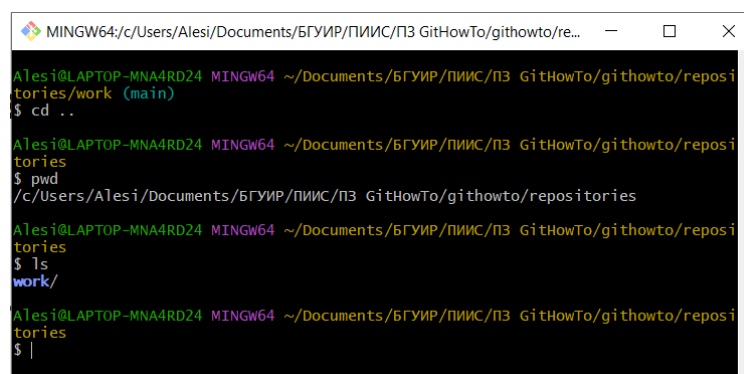
## Часть II: Несколько репозиториев

### Задание 30. Клонирование репозиториев

Если вы работаете в команде, последующие 8 уроков довольно важны в понимании, т.к. вы почти всегда будете работать с клонированными репозиториями.

Перейдите в директорию repositories:

```
cd ..
pwd
ls
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ cd ..

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$ pwd
/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$ ls
work/

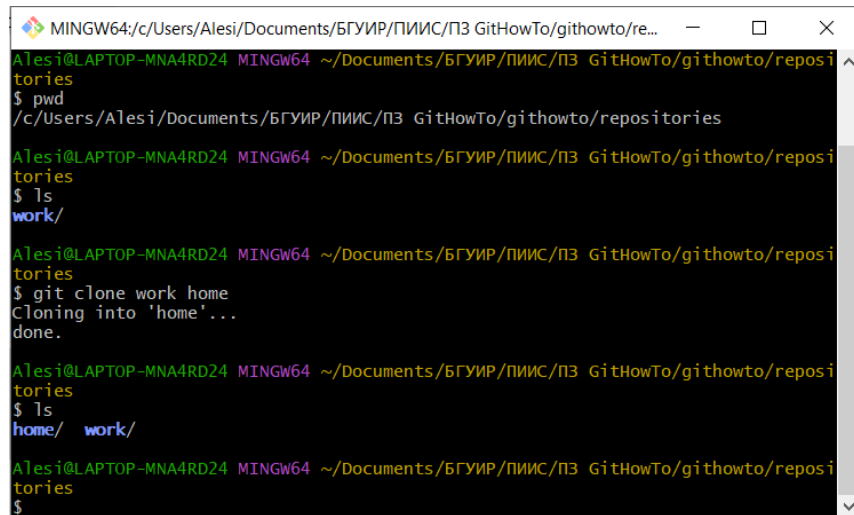
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$
```

Рисунок 95 – Переход в директорию repositories

В этот момент вы должны находиться в директории `repositories`. Здесь должен быть единственный репозиторий под названием `work`.

Давайте создадим клон репозитория.

```
git clone work home
ls
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
$ pwd
/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
$ ls
work/
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
$ git clone work home
Cloning into 'home'...
done.
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
$ ls
home/  work/
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
$
```

Рисунок 96 – Создание клона репозитория `work`

В вашем списке репозиториях теперь должно быть два репозитория: оригинальный репозиторий `work` и клонированный репозиторий `home`.

### Задание 31. Просмотр клонированного репозитория

Давайте взглянем на клонированный репозиторий

```
cd home
ls
```



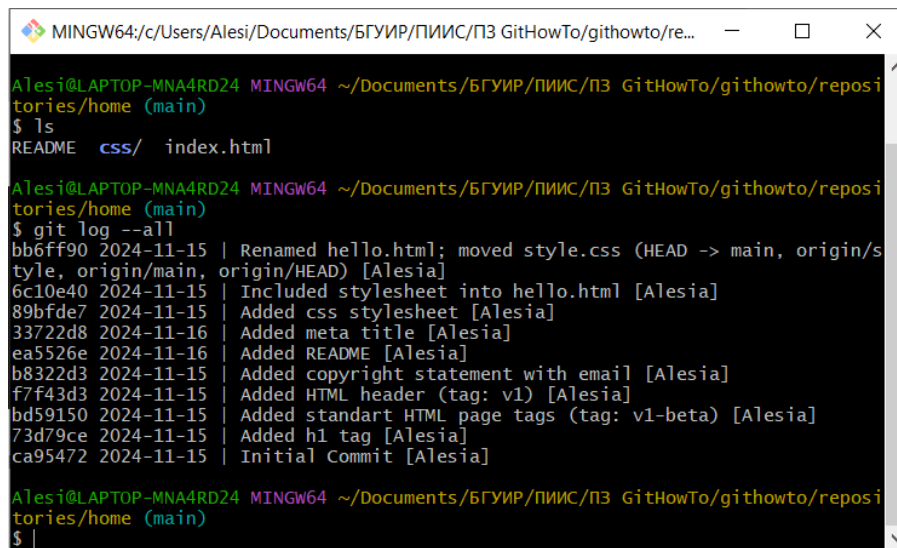
```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories
$ cd home
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$ ls
README  css/  index.html
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$ |
```

Рисунок 97 – Просмотр клонированного репозитория

Вы увидите список всех файлов на верхнем уровне оригинального репозитория `README`, `index.html` и `css`).

Просмотрите историю репозитория:

```
git log --all
```



```
MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
ories/home (main)
$ ls
README  css/  index.html
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
ories/home (main)
$ git log --all
bb6ff90 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> main, origin/s
tyle, origin/main, origin/HEAD) [Alesia]
6c10e40 2024-11-15 | Included stylesheet into hello.html [Alesia]
89bfde7 2024-11-15 | Added css stylesheet [Alesia]
33722d8 2024-11-16 | Added meta title [Alesia]
ea5526e 2024-11-16 | Added README [Alesia]
b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
ories/home (main)
$ |
```

Рисунок 98 – Просмотр истории клонированного репозитория

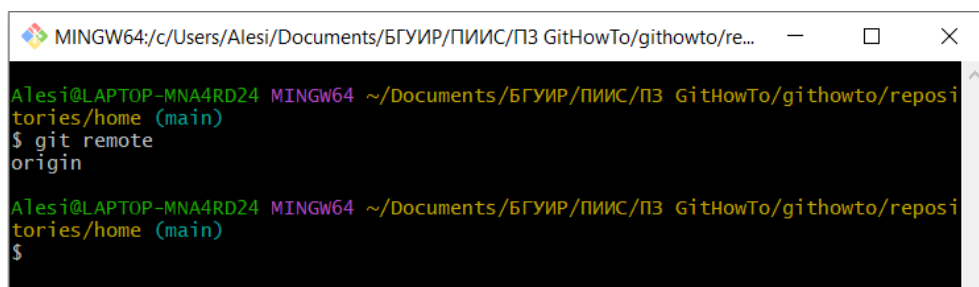
Вы увидите список всех коммитов в новый репозиторий, и он должен совпадать с историей коммитов в оригинальном репозитории. Единственная разница должна быть в названиях веток.

Вы увидите ветку main (HEAD) в списке истории. Вы также увидите ветки со странными именами (origin/main, origin/style и origin/HEAD). Мы поговорим о них чуть позже.

### Задание 32. Что такое origin?

Цель – узнать об именах удаленных репозиториях.

```
git remote
```



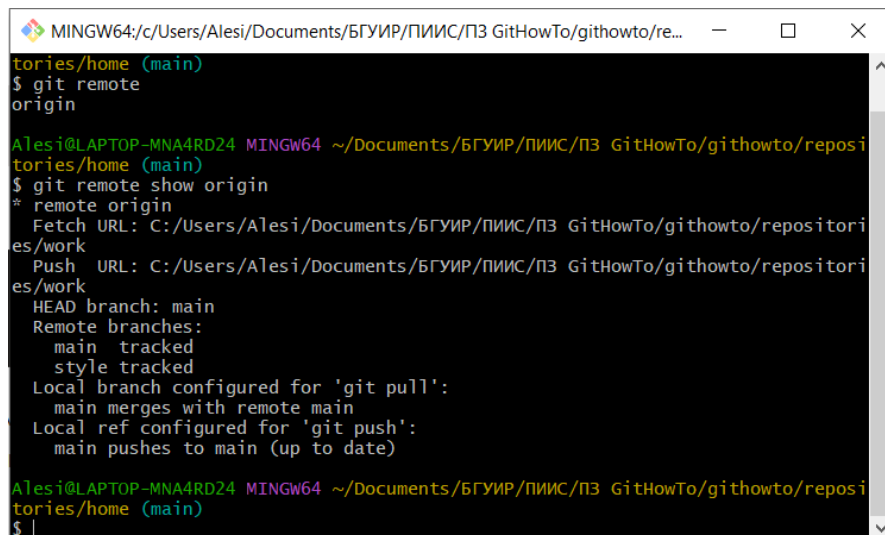
```
MINGW64; c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
ories/home (main)
$ git remote
origin
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
ories/home (main)
$
```

Рисунок 99 – Просмотр удаленного репозитория

Мы видим, что клонированный репозиторий знает об имени по умолчанию удаленного репозитория. Давайте посмотрим, можем ли мы получить более подробную информацию об имени по умолчанию:

```
git remote show origin
```





```
MINGW64/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
tores/home (main)
$ git remote
origin

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tores/home (main)
$ git remote show origin
* remote origin
  Fetch URL: C:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositori
es/work
  Push URL: C:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositori
es/work
  HEAD branch: main
  Remote branches:
    main tracked
    style tracked
  Local branch configured for 'git pull':
    main merges with remote main
  Local ref configured for 'git push':
    main pushes to main (up to date)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tores/home (main)
$
```

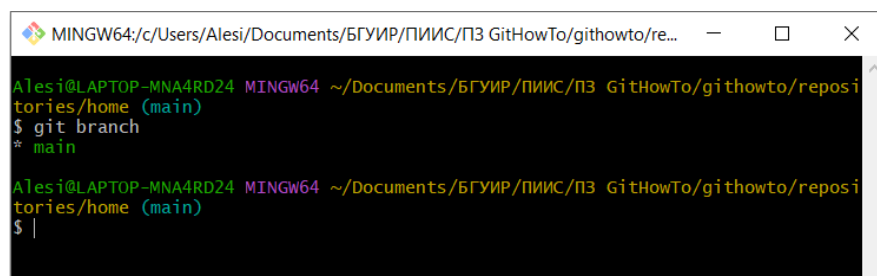
Рисунок 100 – Просмотр информации о репозитории

Мы видим, что имя по умолчанию (origin) удаленного репозитория — изначальное work. Удаленные репозитории обычно размещаются на отдельной машине, возможно, централизованном сервере. Однако, как мы видим здесь, они могут с тем же успехом указывать на репозиторий на той же машине. Нет ничего особенного в имени origin, однако существует традиция использовать origin в качестве имени первичного централизованного репозитория (если таковой имеется).

### Задание 33. Удаленные ветки

Давайте посмотрим на ветки, доступные в нашем клонированном репозитории.

```
git branch
```



```
MINGW64/c:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tores/home (main)
$ git branch
* main

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tores/home (main)
$
```

Рисунок 101 – Просмотр доступных веток

Как мы видим, в списке только ветка main. Где ветка style? Команда git branch выводит только список локальных веток по умолчанию.

Для того чтобы увидеть все ветки, попробуйте следующую команду:

```
git branch -a
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ git branch
* main

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ git branch -a
* main
  remotes/origin/HEAD -> origin/main
  remotes/origin/main
  remotes/origin/style

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$
```

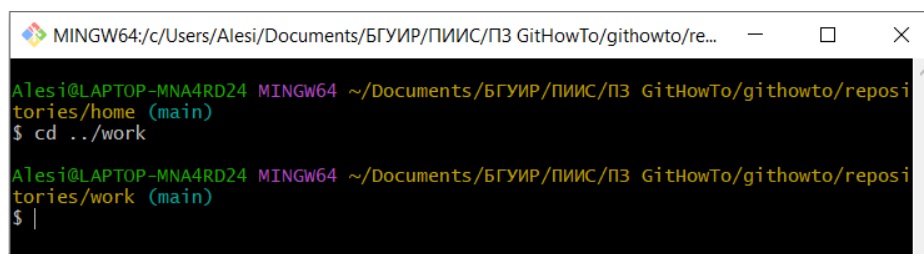
Рисунок 102 – Просмотр всех веток

Git выводит все коммиты в оригинальный репозиторий, но ветки в удаленном репозитории не рассматриваются как локальные. Если мы хотим иметь собственную ветку style, мы должны сами ее создать. Через минуту вы увидите, как это делается.

### Задание 34. Изменение оригинального репозитория

Внесите изменения в оригинальный репозиторий work:

```
cd ../work
```



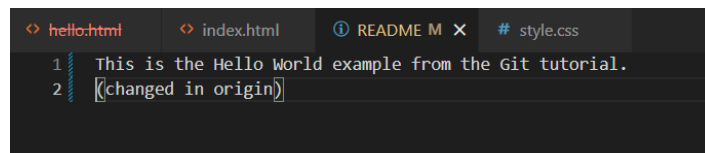
```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repo...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ cd ../work

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ |
```

Рисунок 103 – Переход в репозиторий work

Внесите следующие изменения в файл README:

```
This is the Hello World example from the Git tutorial.
(changed in origin)
```



```
hello.html index.html README M x # style.css
1 This is the Hello World example from the Git tutorial.
2 [(changed in origin)]
```

Рисунок 104 – Изменение файла README

Теперь добавьте это изменение и сделайте коммит:

```
git add README
git commit -m "Changed README in original repo"
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/home (main)
$ cd ../work

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git add README

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ git commit -m "Changed README in original repo"
[main e99726f] Changed README in original repo
1 file changed, 2 insertions(+), 1 deletion(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$
```

Рисунок 105 – Добавление изменения в Git и коммит

Теперь в оригинальном репозитории есть более поздние изменения, которых нет в клонированной версии. Далее мы подтянем и сольем эти изменения в клонированный репозиторий.

### Задание 35. Подтягивание изменений

Цель – научиться подтягивать изменения из удаленного репозитория.

```
cd ../home
git fetch
git log --all
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/work (main)
$ cd ../home

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/home (main)
$ 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), 369 bytes | 18.00 KiB/s, done.
From C:/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/work
   bb6ff90..e99726f  main       -> origin/main

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/home (main)
$ git log --all
e99726f 2024-11-16 | Changed README in original repo (origin/main, origin/HEAD) [Alesia]
bb6ff90 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> main, origin/s
tyle) [Alesia]
6c10e40 2024-11-15 | Included stylesheet into hello.html [Alesia]
89bfde7 2024-11-15 | Added css stylesheet [Alesia]
33722d8 2024-11-16 | Added meta title [Alesia]
ea5526e 2024-11-16 | Added README [Alesia]
b8322d3 2024-11-15 | Added copyright statement with email [Alesia]
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]
73d79ce 2024-11-15 | Added h1 tag [Alesia]
ca95472 2024-11-15 | Initial Commit [Alesia]

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/reposi
tories/home (main)
$ |
```

Рисунок 106 – Подтягивание изменений из удаленного репозитория

На данный момент в репозитории есть все коммиты из оригинального репозитория, но они не интегрированы в локальные ветки клонированного репозитория.

В истории выше найдите коммит «Changed README in original repo». Обратите внимание, что коммит включает в себя коммиты origin/main и origin/HEAD.

Теперь давайте посмотрим на коммит «Renamed hello.html; moved style.css». Вы увидите, что локальная ветка main указывает на этот коммит, а не на новый коммит, который мы только что подтянули.

Выводом является то, что команда `git fetch` будет подтягивать новые коммиты из удаленного репозитория, но не будет сливать их с вашими работами в локальных ветках.

Мы можем продемонстрировать, что клонированный файл README не изменился.

```
cat README
```

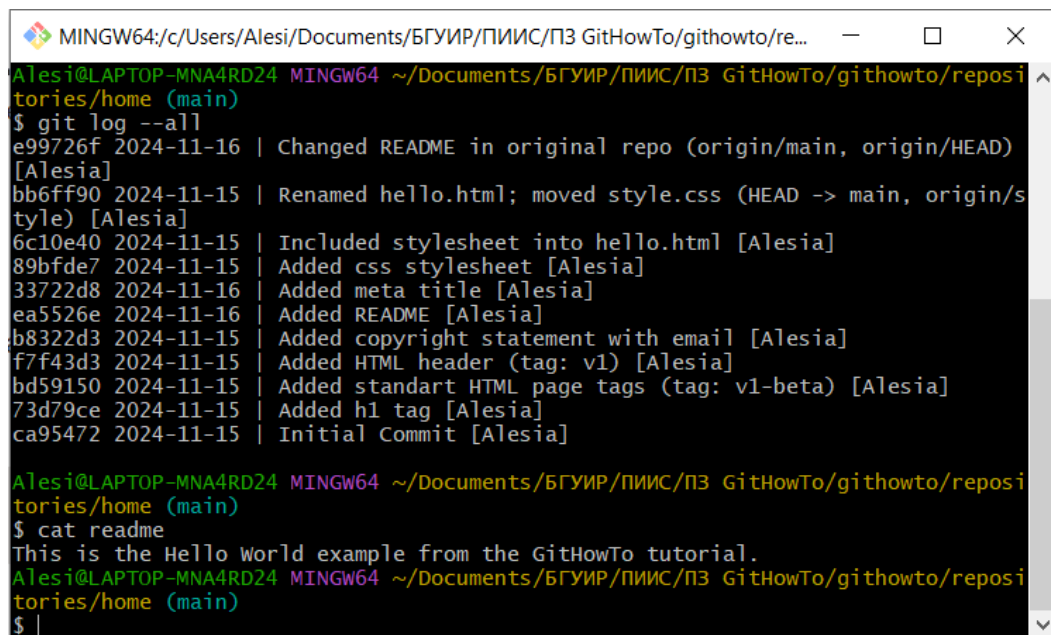
A screenshot of a terminal window with a dark background. The window title is 'MINGW64; c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/re...'. The terminal shows the following commands and output:  
\$ git log --all  
e99726f 2024-11-16 | Changed README in original repo (origin/main, origin/HEAD) [Alesia]  
bb6ff90 2024-11-15 | Renamed hello.html; moved style.css (HEAD -> main, origin/s  
6c10e40 2024-11-15 | Included stylesheet into hello.html [Alesia]  
89bfde7 2024-11-15 | Added css stylesheet [Alesia]  
33722d8 2024-11-16 | Added meta title [Alesia]  
ea5526e 2024-11-16 | Added README [Alesia]  
b8322d3 2024-11-15 | Added copyright statement with email [Alesia]  
f7f43d3 2024-11-15 | Added HTML header (tag: v1) [Alesia]  
bd59150 2024-11-15 | Added standart HTML page tags (tag: v1-beta) [Alesia]  
73d79ce 2024-11-15 | Added h1 tag [Alesia]  
ca95472 2024-11-15 | Initial Commit [Alesia]  
  
\$ cat readme  
This is the Hello World example from the GitHowTo tutorial.  
\$

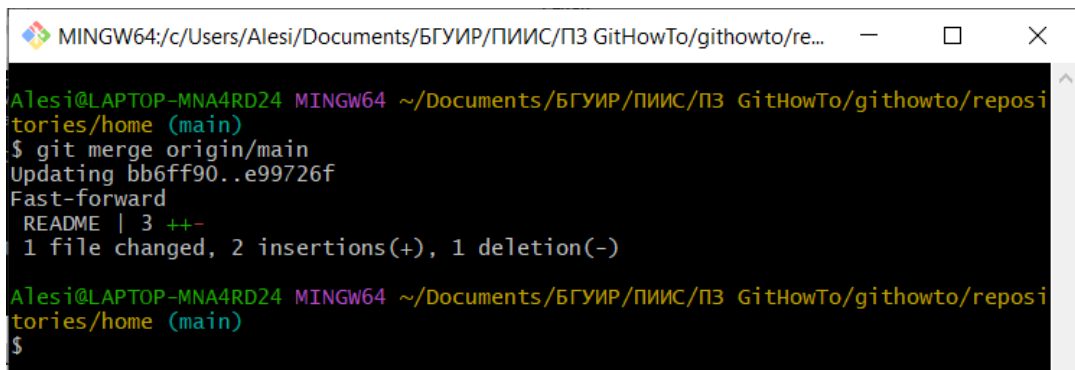
Рисунок 107 – Просмотр README

Как видите, никаких изменений.

### Задание 36. Слияние подтянутых изменений

Слейте подтянутые изменения в локальную ветку main.

```
git merge origin/main
```



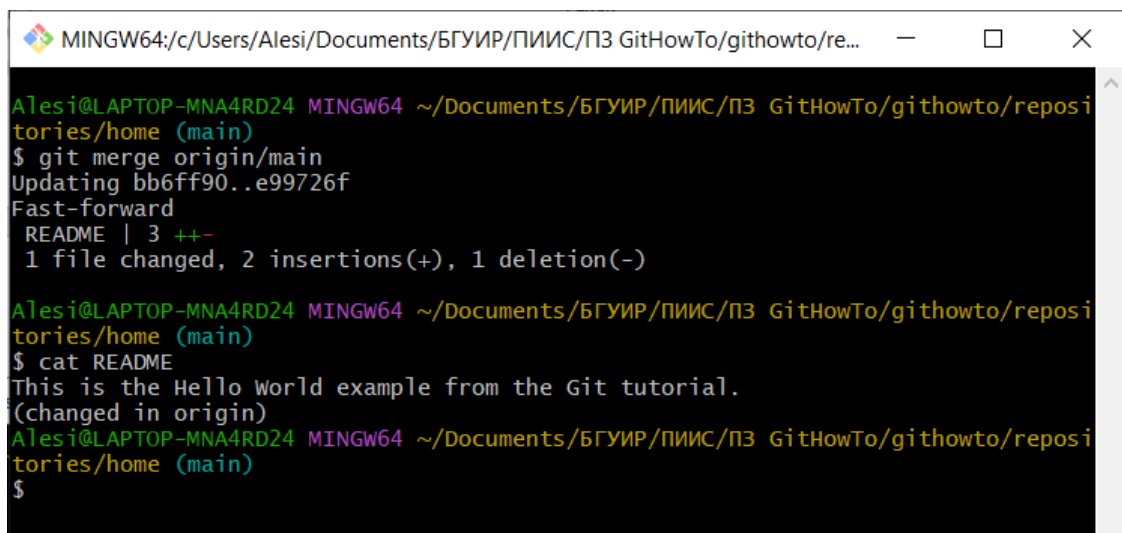
```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
ories/home (main)
$ git merge origin/main
Updating bb6ff90..e99726f
Fast-forward
 README | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
ories/home (main)
$
```

Рисунок 108 – Слияние подтянутых изменений в локальную ветку main

Еще раз проверьте файл README. Сейчас мы должны увидеть изменения.

```
cat README
```



```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
ories/home (main)
$ git merge origin/main
Updating bb6ff90..e99726f
Fast-forward
 README | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
ories/home (main)
$ cat README
This is the Hello World example from the Git tutorial.
(changed in origin)
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghithowto/reposi
ories/home (main)
$
```

Рисунок 109 – Проверка файла README

Вот и изменения. Хотя команда `git fetch` не сливает изменения, мы можем вручную слить изменения из удаленного репозитория.

Команда `fetch` позволяет контролировать то, что именно подтягивается и сливается в ваши локальные ветки, но для удобства существует также команда `pull`, которая подтягивает и сливает изменения из удаленной ветки в текущую одним вызовом.

```
git pull
```

...эквивалентна следующим двум шагам:

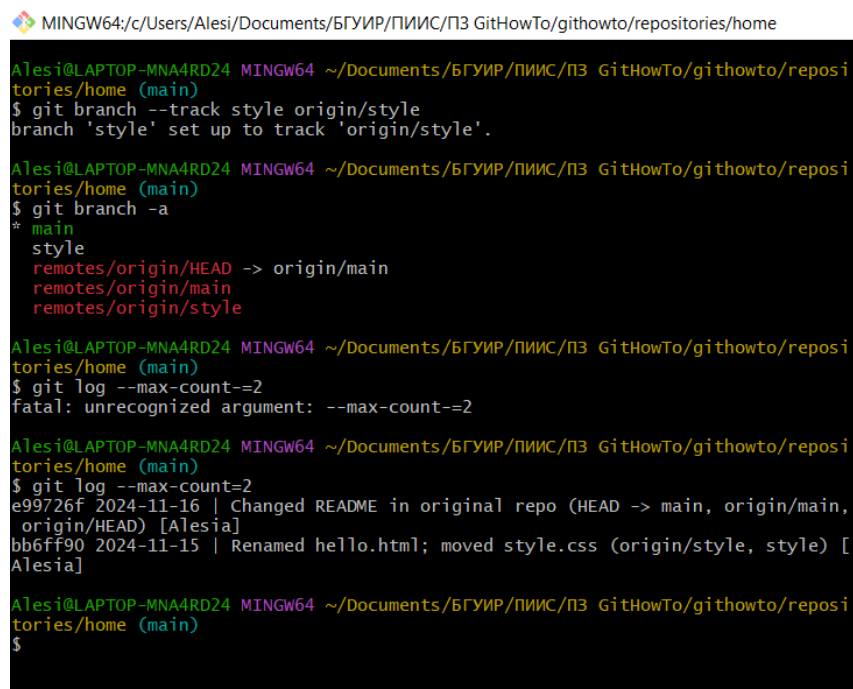
```
git fetch
git merge origin/main
```

### Задание 37. Добавление ветки наблюдения

Ветки, которые начинаются с `remotes/origin` являются ветками оригинального репозитория. Обратите внимание, что у вас больше нет ветки под названием `style`, но система контроля версий знает, что в оригинальном репозитории ветка `style` была.

Добавьте локальную ветку, которая отслеживает удаленную ветку:

```
git branch --track style origin/style
git branch -a
git log --max-count=2
```



```
MINGW64/c:/Users/Alesia/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home
Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$ git branch --track style origin/style
branch 'style' set up to track 'origin/style'.

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$ git branch -a
* main
  style
  remotes/origin/HEAD -> origin/main
  remotes/origin/main
  remotes/origin/style

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$ git log --max-count=2
fatal: unrecognized argument: --max-count=2

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$ git log --max-count=2
e99726f 2024-11-16 | Changed README in original repo (HEAD -> main, origin/main, origin/HEAD) [Alesia]
bb6ff90 2024-11-15 | Renamed hello.html; moved style.css (origin/style, style) [Alesia]

Alesia@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/githowto/repositories/home (main)
$
```

Рисунок 110 – Добавление локальной ветки для отслеживания удаленной

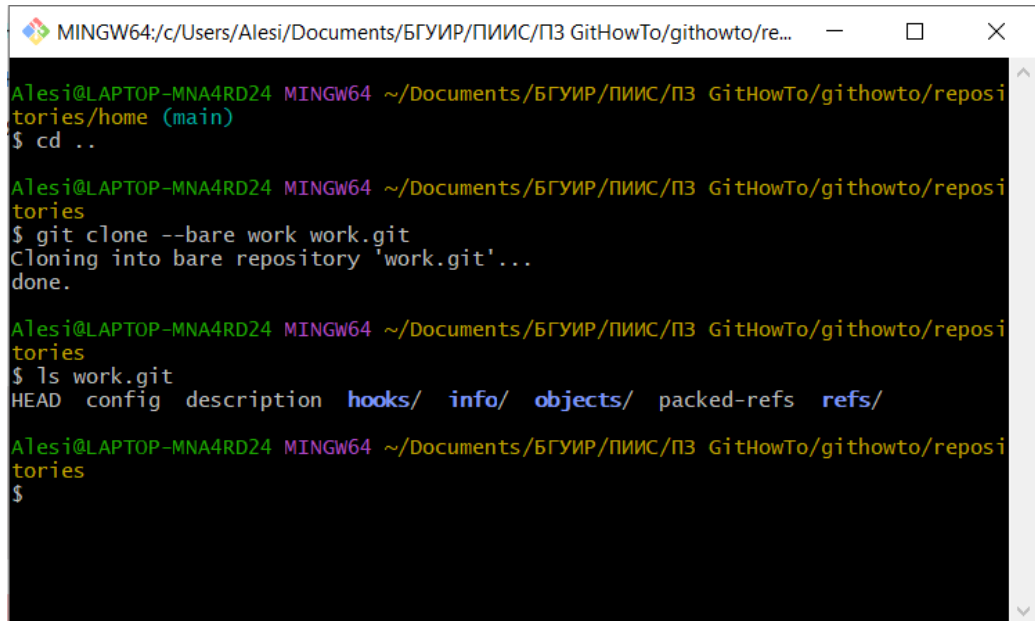
Теперь мы можем видеть ветку `style` в списке веток и логе.

### Задание 38. Чистые репозитории

Чистый репозиторий — это репозиторий, не имеющий рабочей директории. Он содержит только директорию `.git`, в которой Git хранит все свои внутренние данные. Основное предназначение таких репозиторий — быть центральным хранилищем, в которое разработчики могут отправлять и из которого могут получать данные. Поэтому в них нет смысла создавать рабочие файлы, они будут только впустую занимать место на диске. Чистые репозитории также используются в сервисах Git-хостинга таких, как GitHub и GitLab. В следующих уроках мы узнаем, как создать чистый репозиторий и как отправлять в него изменения.

Создайте чистый репозиторий:

```
cd ..
git clone --bare work work.git
ls work.git
```



```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/home (main)
$ cd ..

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$ git clone --bare work work.git
Cloning into bare repository 'work.git'...
done.

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$ ls work.git
HEAD config description hooks/ info/ objects/ packed-refs refs/

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$
```

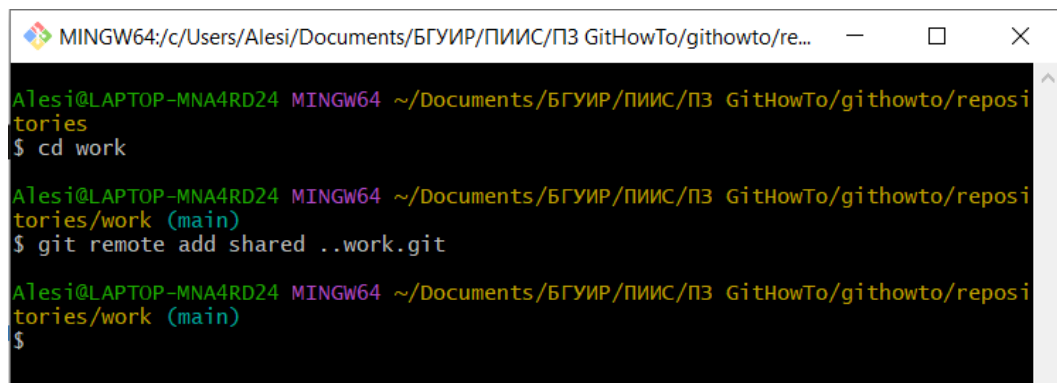
Рисунок 111 – Создание чистого репозитория

Принято считать, что репозитории, заканчивающиеся на .git, являются чистыми репозиториями. Мы видим, что в репозитории work.git нет рабочей директории. По сути, это просто директория .git из обычного репозитория.

### Задание 39. Добавление удаленного репозитория

Давайте добавим репозиторий work.git к нашему оригинальному репозиторию.

```
cd work
git remote add shared ../work.git
```



```
MINGW64:/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories
$ cd work

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$ git remote add shared ../work.git

Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/reposi
tories/work (main)
$
```

Рисунок 112 – Добавление удаленного репозитория

## Задание 40. Отправка изменений

Поскольку чистые репозитории обычно располагаются на каком-либо удаленном сервере, вы не сможете туда просто зайти, дабы подтянуть изменения. Поэтому нам необходимо как-нибудь передать наши изменения в репозиторий.

Начнем с создания изменения, которое нужно передать в репозиторий. Отредактируйте README и закоммитьте его:

```
This is the Hello World example from the Git tutorial.  
(changed in the origin and pushed to shared)  
  
git switch main  
git add README  
git commit -m "Added shared comment to readme"
```

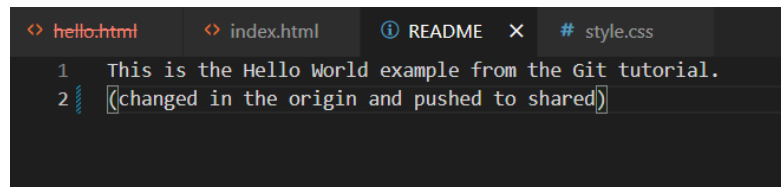


Рисунок 113 – Редактирование README

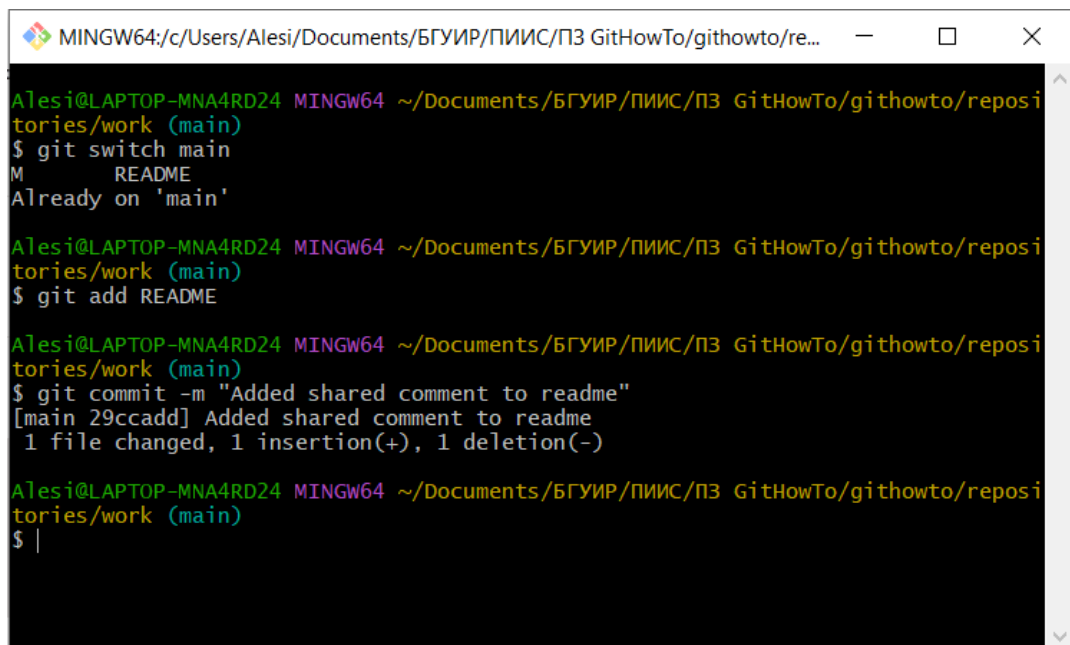


Рисунок 114 – Коммит изменений

Теперь отправьте изменения в общий репозиторий.

```
git push shared main
```



```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/re...
Already on 'main'
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git add README
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git commit -m "Added shared comment to readme"
[main 29ccadd] Added shared comment to readme
1 file changed, 1 insertion(+), 1 deletion(-)
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ git push shared main
fatal: '..work.git' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$
```

Рисунок 115 – Отправка изменений в общий репозиторий

Общим называется репозиторий, получающий отправленные нами изменения. Помните, мы добавили его в качестве удаленного репозитория в предыдущем уроке?

#### Задание 41. Подтягивание общих изменений

Быстро переключитесь в репозиторий home и подтяните изменения, только что отправленные в общий репозиторий.

```
cd ../home
```

Продолжите с...

```
git remote add shared ../work.git
git branch --track shared main
git pull shared main
cat README
```

```
MINGW64/c/Users/Alesi/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/work (main)
$ cd ../home
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ git remote add shared ../work.git
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ git branch --track shared main
branch 'shared' set up to track 'main'.
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ git pull shared main
From ../work
 * branch          main      -> FETCH_HEAD
 * [new branch]    main      -> shared/main
Already up to date.
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$ cat README
This is the Hello World example from the Git tutorial.
(changed in origin)
Alesi@LAPTOP-MNA4RD24 MINGW64 ~/Documents/БГУИР/ПИИС/ПЗ GitHowTo/ghowto/repositories/home (main)
$
```

Рисунок 116 – Подтягивание общих изменений

## Задание 42. Размещение ваших Git-репозитория

Хотите создать свой собственный GitHub? Существует множество способов совместного использования репозитория Git по сети. Здесь приведен простой и быстрый (но ненадежный и опасный) способ.

Запуск Git-сервера:

```
# (From the "repositories" directory)
git daemon --verbose --export-all --base-path=.
```

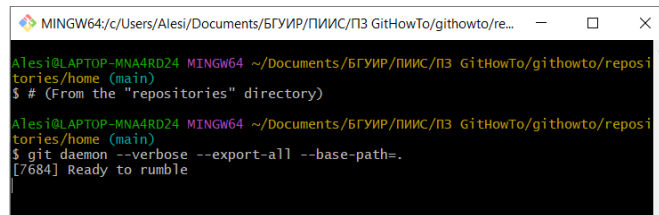


Рисунок 117 – Запуск Git-сервера

Теперь в отдельном окне терминала перейдите в вашу директорию repositories:

```
# (From the "repositories" directory)
git clone git://localhost/work.git network_work
cd network_work
ls
```

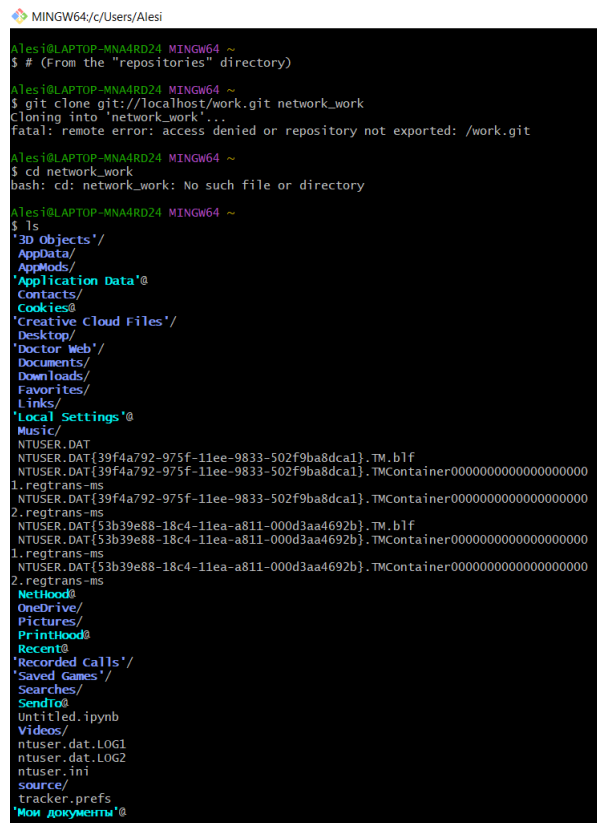


Рисунок 118 – Открытие Git-сервера

Вы увидите копию проекта work.

Если вы хотите разрешить отправку изменений (push) в репозиторий Git Daemon, добавьте метку `--enable=receive-pack` к команде `git daemon`. Будьте осторожны, этот сервер не производит аутентификацию, поэтому любой сможет отправлять изменения в ваш репозиторий.

На этом этапе вам открываются безграничные возможности. Смелее! Возьмите в аренду сервер, купите доменное имя, разместите на этом сервере свои репозитории и наслаждайтесь своим личным GitHub!

Если серьезно, то вы можете самостоятельно разместить свой личный сервер GitLab. Этот продукт бесплатный и с открытым исходным кодом.

*Спасибо за использование GitHowTo! Надеюсь, вам было интересно.*

**Выводы:** Git — мощная и сложная распределенная система контроля версий, которая работает с изменениями, а не файлами. Git позволяет создавать репозитории с ветками, коммитить и отслеживать изменения в файлах, а также позволяет откатываться к любой интересующей версии проекта благодаря истории изменений.