

# Отчет по Лабораторной Работе №1

Работа с Git

Озьяс Стев Икнэль Дани

# Содержание

1	Цель работы	6
2	Задание	7
3	Теоретическое введение	8
4	Выполнение лабораторной работы	9
4.1	Подготовка	9
4.2	Создание проекта	9
4.3	Внесение изменений	10
4.4	История	12
4.5	Получение старых версий	13
4.6	Создание тегов версий	14
4.7	Отмена локальных изменений (до индексации)	16
4.8	Отмена проиндексированных изменений (перед коммитом)	16
4.9	Отмена коммитов	17
4.10	Удаление коммитов из ветки	18
4.11	Удаление тега oops	19
4.12	Изменение предыдущего коммита	20
4.13	Перемещение файлов	20
4.14	Подробнее о структуре	21
4.15	Git внутри: Каталог .git	22
4.16	Работа непосредственно с объектами git	23
4.17	Создание ветки	24
4.18	Навигация по веткам	25
4.19	Изменения в ветке master	26
4.20	Слияние	26
4.21	Создание конфликта	27
4.22	Разрешение конфликтов	27
4.23	Сброс ветки style	27
4.24	Сброс ветки master	28
4.25	Перебазирование	29
4.26	Слияние в ветку master	29
4.27	Клонирование репозитория	30
4.28	Что такое origin?	30
4.29	Удаленные ветки	31
4.30	Изменение оригинального репозитория	31

4.31 Слияние извлеченных изменений . . . . .	32
4.32 Добавление ветки наблюдения . . . . .	32
4.33 Создание чистого репозитория . . . . .	33
4.34 Отправка и извлечение изменений . . . . .	33
5 Выводы	35
Список литературы	36

## Список иллюстраций

4.1	Настройка git . . . . .	9
4.2	Создание репозитория . . . . .	10
4.3	Внесение изменений в содержимое репозитория . . . . .	11
4.4	Внесение нескольких изменений в содержимое репозитория . . .	12
4.5	Просмотр истории . . . . .	13
4.6	Просмотр разных версий репозитория . . . . .	14
4.7	Создание тегов версий . . . . .	15
4.8	Переключение по имени тега и просмотр доступных тегов . . . .	15
4.9	Отмена локальных изменений (до индексации) . . . . .	16
4.10	Отмена проиндексированных изменений (перед коммитом) . . .	17
4.11	Отмена коммитов . . . . .	18
4.12	Удаление коммитов из ветки . . . . .	19
4.13	Удаление тега оорс . . . . .	19
4.14	Изменение предыдущего коммита . . . . .	20
4.15	Перемещение файлов . . . . .	21
4.16	Добавление нового файла в репозиторий . . . . .	21
4.17	Добавление нового файла в репозиторий . . . . .	22
4.18	Каталог .git . . . . .	23
4.19	Работа непосредственно с объектами git . . . . .	23
4.20	Поиск оригинального файла hello.html . . . . .	24
4.21	Создание ветки . . . . .	24
4.22	Просмотр логов новой ветки . . . . .	25
4.23	Переключение между ветками . . . . .	25
4.24	Изменения в ветке master . . . . .	26
4.25	Просмотр веток . . . . .	26
4.26	Слияние веток . . . . .	26
4.27	Создание конфликта . . . . .	27
4.28	Разрешение конфликта . . . . .	27
4.29	Поиск коммита перед слиянием . . . . .	28
4.30	Сброс ветки style . . . . .	28
4.31	Поиск коммита перед конфликтом . . . . .	28
4.32	Сброс ветки master . . . . .	29
4.33	Перебазирование . . . . .	29
4.34	Слияние style в master . . . . .	29
4.35	Клонирование репозитория . . . . .	30
4.36	Просмотр имени по умолчанию удаленного репозитория . . . .	31
4.37	Просмтр доступных веток . . . . .	31

4.38	Изменение оригинального репозитория . . . . .	31
4.39	Извлечение изменений . . . . .	32
4.40	Слияние извлеченных изменений . . . . .	32
4.41	Добавление ветки наблюдения . . . . .	33
4.42	Создание чистого репозитория . . . . .	33
4.43	Отправка и извлечение изменений . . . . .	34

# 1 Цель работы

Приобретение практических навыков работы с ситемой управления версиями Git.

## 2 Задание

Провести базовую настройку гит, создать проект и, используя его, изучить основные действия с репозиторием.

### 3 Теоретическое введение

Git — распределённая система управления версиями. Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux, первая версия выпущена 7 апреля 2005 года; координатор — Дзюн Хамано [1].

Работа выполнена с помощью веб-сервиса GitHub [2].



## 4 Выполнение лабораторной работы

### 4.1 Подготовка

Сначала настроим `core.autocrlf` с параметрами `true` и `input`, чтобы сделать все переводы строк текстовых файлов в главном репозитории одинаковыми, а затем настроим отображение `unicode`(рис. 4.1).

```
dacossti@DESKTOP-124TCDC:~$ git config --global user.name "Dacossti"
dacossti@DESKTOP-124TCDC:~$ git config --global user.email "osiasstavel5@gmail.com"
dacossti@DESKTOP-124TCDC:~$ git config --global core.autocrlf input
dacossti@DESKTOP-124TCDC:~$ git config --global core.safecrlf true
dacossti@DESKTOP-124TCDC:~$ git config --global core.quotepath off
```

Рис. 4.1: Настройка git

### 4.2 Создание проекта

Создадим пустой каталог `hello`, а в нём файл с именем `hello.html`. Затем создадим git репозиторий из этого каталога, выполнив команду `git init`. Добавим файл в репозиторий и проверим статус, который сообщает, что коммитить нечего(рис. 4.2).

```

dacossti@DESKTOP-124TCDC:~$ mkdir hello
dacossti@DESKTOP-124TCDC:~$ cd hello
dacossti@DESKTOP-124TCDC:~/hello$ touch hello.html
dacossti@DESKTOP-124TCDC:~/hello$ echo "Hello, World!" > hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/dacossti/hello/.git/
dacossti@DESKTOP-124TCDC:~/hello$ git add hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git commit -m "Initial Commit"
[master (root-commit) cd073bf] Initial Commit
 1 file changed, 1 insertion(+)
 create mode 100644 hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git status
On branch master
nothing to commit, working tree clean

```

Рис. 4.2: Создание репозитория

## 4.3 Внесение изменений

Изменим содержимое файла hello.html на:

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

Проверив состояние рабочего каталога увидим, что git знает, что файл hello.html был изменен, но при этом эти изменения еще не зафиксированы в репозитории. Теперь проиндексируем изменения и снова посмотрим статус, в нём указано, что изменения пока не записаны в репозиторий. И наконец закоммитим изменения, внеся их в репозиторий и снова посмотрим статус, который теперь показывает, что все изменения внесены в репозиторий (рис. 4.3).

```
dacossti@DESKTOP-124TDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TDC:~/hello$ cat hello.html
<h1>Hello, World!</h1>
dacossti@DESKTOP-124TDC:~/hello$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   hello.html

no changes added to commit (use "git add" and/or "git commit -a")
dacossti@DESKTOP-124TDC:~/hello$ git add hello.html
dacossti@DESKTOP-124TDC:~/hello$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html
```

Рис. 4.3: Внесение изменений в содержимое репозитория

Изменим страницу «Hello, World», чтобы она содержала стандартные теги

и

.

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

Теперь добавим это изменение в индекс git и добавим заголовки HTML (секцию ) к странице «Hello, World». Проверив текущий статус увидим, что hello.html указан дважды в состоянии. Первое изменение (добавление стандартных тегов) проиндексировано и готово к коммиту. Второе изменение (добавление заголовков HTML) является непроиндексированным. Произведем коммит проиндексированного изменения, затем проиндексируем оставшееся изменение, посмотрим статус и прокоммитим его(рис. 4.4).

```

dacossti@DESKTOP-124TCDC:~/hello$ git commit
[master b6c093e] «Added h1 tag»
1 file changed, 1 insertion(+), 1 deletion(-)
dacossti@DESKTOP-124TCDC:~/hello$ git status
On branch master
nothing to commit, working tree clean
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ cat hello.html
<h1>Hello, World!</h1>
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git add hello.html
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ cat hello.html
<html>
<head>
</head>
<body>
<h1>Hello, World!</h1>
</body>
</html>
dacossti@DESKTOP-124TCDC:~/hello$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

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

dacossti@DESKTOP-124TCDC:~/hello$ git commit -m "Added standard HTML page tags"
[master b50bc7f] Added standard HTML page tags
1 file changed, 4 insertions(+)
dacossti@DESKTOP-124TCDC:~/hello$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   hello.html

```

Рис. 4.4: Внесение нескольких изменений в содержимое репозитория

## 4.4 История

Посмотрим список произведённых изменений в стандартном виде, затем в однострочном, а также с указанием времени и количества (рис. 4.5).

```

dacossti@DESKTOP-124TDCD:~/hello$ git log
commit 1a44782cac245a2654d36f7f63a06a45618bf504 (HEAD -> master)
Author: Dacossti <osiasstavel15@gmail.com>
Date:   Wed May 28 23:31:25 2025 +0300

    Added HTML header

commit b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6
Author: Dacossti <osiasstavel15@gmail.com>
Date:   Wed May 28 23:30:25 2025 +0300

    Added standard HTML page tags

commit b6c093ed48b86b145559c4b0571cbae4ae452127
Author: Dacossti <osiasstavel15@gmail.com>
Date:   Wed May 28 23:24:22 2025 +0300

    «Added h1 tag»

commit cd073bfa2b66136c94b6ee7d8ce1a57a89fceff5
Author: Dacossti <osiasstavel15@gmail.com>
Date:   Wed May 28 23:20:57 2025 +0300

    Initial Commit
dacossti@DESKTOP-124TDCD:~/hello$ git log --pretty=oneline
1a44782cac245a2654d36f7f63a06a45618bf504 (HEAD -> master) Added HTML header
b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6 Added standard HTML page tags
b6c093ed48b86b145559c4b0571cbae4ae452127 «Added h1 tag»
cd073bfa2b66136c94b6ee7d8ce1a57a89fceff5 Initial Commit
dacossti@DESKTOP-124TDCD:~/hello$ git log --pretty=oneline --max-count=2
1a44782cac245a2654d36f7f63a06a45618bf504 (HEAD -> master) Added HTML header
b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6 Added standard HTML page tags
dacossti@DESKTOP-124TDCD:~/hello$ git log --pretty=oneline --since='5 minutes ago'
1a44782cac245a2654d36f7f63a06a45618bf504 (HEAD -> master) Added HTML header
b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6 Added standard HTML page tags
dacossti@DESKTOP-124TDCD:~/hello$ git log --pretty=oneline --until='5 minutes ago'
b6c093ed48b86b145559c4b0571cbae4ae452127 «Added h1 tag»
cd073bfa2b66136c94b6ee7d8ce1a57a89fceff5 Initial Commit
dacossti@DESKTOP-124TDCD:~/hello$ git log --pretty=oneline --author=<Dacossti>
-bash: syntax error near unexpected token `newline'
dacossti@DESKTOP-124TDCD:~/hello$ git log --pretty=oneline --author=Dacossti

```

Рис. 4.5: Просмотр истории

## 4.5 Получение старых версий

Изучим данные лога и найдем там хэш первого коммита, используя его вернемся к первой версии и просмотрим файл `hello.html`, действительно, увидим первую версию. Затем вернемся к последней версии в ветке `master` и вновь посмотрим на файл (рис. 4.6).

```
dacossti@DESKTOP-124TDC:~/hello$ git checkout b50bc7f
Note: switching to 'b50bc7f'.

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 b50bc7f Added standard HTML page tags
dacossti@DESKTOP-124TDC:~/hello$ cat hello.html
<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
dacossti@DESKTOP-124TDC:~/hello$ git checkout master
Previous HEAD position was b50bc7f Added standard HTML page tags
Switched to branch 'master'
```

Рис. 4.6: Просмотр разных версий репозитория

## 4.6 Создание тегов версий

Назовем текущую версию страницы hello первой (v1). Создадим тег первой версии и используем его для того чтобы вернуться к предыдущей, которой также присвоим тег(рис. 4.7).

```

dacossti@DESKTOP-124TCDC:~/hello$ git tag v1
dacossti@DESKTOP-124TCDC:~/hello$ 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 b50bc7f Added standard HTML page tags
dacossti@DESKTOP-124TCDC:~/hello$ cat hello.html
<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
dacossti@DESKTOP-124TCDC:~/hello$ git tag v1-beta
dacossti@DESKTOP-124TCDC:~/hello$ git checkout v1
Previous HEAD position was b50bc7f Added standard HTML page tags
HEAD is now at 1a44782 Added HTML header
dacossti@DESKTOP-124TCDC:~/hello$ git checkout v1-beta
Previous HEAD position was 1a44782 Added HTML header
HEAD is now at b50bc7f Added standard HTML page tags

```

Рис. 4.7: Создание тегов версий

Переключимся по тегам между двумя отмеченными версиями. Просмотрим все доступные теги(их два) и посмотрим теки в логе(рис. 4.8).

```

dacossti@DESKTOP-124TCDC:~/hello$ git checkout v1-beta
Previous HEAD position was 1a44782 Added HTML header
HEAD is now at b50bc7f Added standard HTML page tags
dacossti@DESKTOP-124TCDC:~/hello$ git tag
v1
v1-beta
dacossti@DESKTOP-124TCDC:~/hello$ git log master --all
commit 1a44782cac245a2654d36f7f63a06a45618bf504 (tag: v1, master)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:31:25 2025 +0300

    Added HTML header

commit b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6 (HEAD, tag: v1-beta)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:30:25 2025 +0300

    Added standard HTML page tags

commit b6c093ed48b86b145559c4b0571cbae4ae452127
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:24:22 2025 +0300

    «Added h1 tag»

commit cd073bfa2b66136c94b6ee7d8ce1a57a89fceff5
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:20:57 2025 +0300

    Initial Commit

```

Рис. 4.8: Переключение по имени тега и просмотр доступных тегов

## 4.7 Отмена локальных изменений (до индексации)

Убедимся, что мы находимся на последнем коммите ветки master и внесем изменение в файл hello.html в виде нежелательного комментария. Затем проверим статус, увидим, что изменения ещё не проиндексированы. Используем команду git checkout для переключения версии файла hello.html в репозитории(рис. 4.9).

```
dacossti@DESKTOP-124TCDC:~/hello$ git checkout master
Previous HEAD position was b50bc7f Added standard HTML page tags
Switched to branch 'master'
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   hello.html

no changes added to commit (use "git add" and/or "git commit -a")
dacossti@DESKTOP-124TCDC:~/hello$ git checkout hello.html
Updated 1 path from the index
dacossti@DESKTOP-124TCDC:~/hello$ git status
On branch master
nothing to commit, working tree clean
dacossti@DESKTOP-124TCDC:~/hello$ cat hello.html
<html>
<head>
</head>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

Рис. 4.9: Отмена локальных изменений (до индексации)

## 4.8 Отмена проиндексированных изменений (перед коммитом)

Внесем 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>
```

Проиндексируем это изменение и проверим состояние. Состояние показывает, что изменение было проиндексировано и готово к коммиту. Используем команду `git reset`, чтобы сбросить буферную зону к HEAD. Это очищает буферную зону от изменений, которые мы только что проиндексировали. И переключимся на последнюю версию коммита, посмотрев статус увидим, что наш каталог опять чист(рис. 4.10).

```
dacossti@DESKTOP-124TDCD:~/hello$ nano hello.html
dacossti@DESKTOP-124TDCD:~/hello$ git add hello.html
dacossti@DESKTOP-124TDCD:~/hello$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   hello.html

dacossti@DESKTOP-124TDCD:~/hello$ git reset HEAD hello.html
Unstaged changes after reset:
M       hello.html
dacossti@DESKTOP-124TDCD:~/hello$ git checkout hello.html
Updated 1 path from the index
dacossti@DESKTOP-124TDCD:~/hello$ git status
On branch master
nothing to commit, working tree clean
```

Рис. 4.10: Отмена проиндексированных изменений (перед коммитом)

## 4.9 Отмена коммитов

Изменим файл `hello.html` на следующий.

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

</html>

Проиндексируем изменения файла и прокоммитим их. Чтобы отменить коммит, нам необходимо сделать коммит, который удаляет изменения, сохраненные нежелательным коммитом. Перейдем в редактор, где изменим нежелательный коммит. Проверим лог. Проверка лога показывает нежелательные и отмененные коммиты в наш репозиторий(рис. 4.11).

```
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git add hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git commit -m "Oops, we didn't want this commit"
[master 8439e8f] Oops, we didn't want this commit
1 file changed, 1 insertion(+)
dacossti@DESKTOP-124TCDC:~/hello$ git revert HEAD
[master f0e4194] Revert "Oops, we didn't want this commit"
1 file changed, 1 deletion(-)
dacossti@DESKTOP-124TCDC:~/hello$ git log
commit f0e41940606cb0c1cdedd75e1e8993d39cf35b16 (HEAD -> master)
Author: Dacossti <osiasstave15@gmail.com>
Date: Wed May 28 23:43:54 2025 +0300

    Revert "Oops, we didn't want this commit"

    This reverts commit 8439e8f381cf78d171b9fba5d4bb50829d34a442.

commit 8439e8f381cf78d171b9fba5d4bb50829d34a442
Author: Dacossti <osiasstave15@gmail.com>
Date: Wed May 28 23:43:40 2025 +0300

    Oops, we didn't want this commit
```

Рис. 4.11: Отмена коммитов

## 4.10 Удаление коммитов из ветки

Удалим последние два коммита с помощью сброса, сначала отметим последний коммит тегом, чтобы его можно было потом найти. Используем команду `git reset`, чтобы вернуться к версии до этих коммитов. Теперь в логе их нет, но если посмотреть логи с опцией `-all` можно всё ещё их увидеть, но метка HEAD находится на нужной нам версии(рис. 4.12).

```

dacossti@DESKTOP-124TDCD:~/hello$ git tag oops
dacossti@DESKTOP-124TDCD:~/hello$ git reset --hard v1
HEAD is now at 1a44782 Added HTML header
dacossti@DESKTOP-124TDCD:~/hello$ git log
commit 1a44782cac245a2654d36f7f63a06a45618bf504 (HEAD -> master, tag: v1)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:31:25 2025 +0300

    Added HTML header

commit b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6 (tag: v1-beta)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:30:25 2025 +0300

    Added standard HTML page tags

commit b6c093ed48b86b145559c4b0571cbae4ae452127
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:24:22 2025 +0300

    «Added h1 tag»

commit cd073bfa2b66136c94b6ee7d8ce1a57a89fceff5
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:20:57 2025 +0300

    Initial Commit

```

Рис. 4.12: Удаление коммитов из ветки

## 4.11 Удаление тега оорс

Удалим тег оорс и коммиты, на которые он ссылался, сборщиком мусора. Теперь этот тег не отображается в репозитории(рис. 4.13).

```

dacossti@DESKTOP-124TDCD:~/hello$ git tag -d oops
Deleted tag 'oops' (was f0e4194)
dacossti@DESKTOP-124TDCD:~/hello$ git log --all
commit 1a44782cac245a2654d36f7f63a06a45618bf504 (HEAD -> master, tag: v1)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:31:25 2025 +0300

    Added HTML header

commit b50bc7f8d377c23a73bc82a23f7d94b5f7b113b6 (tag: v1-beta)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:30:25 2025 +0300

    Added standard HTML page tags

commit b6c093ed48b86b145559c4b0571cbae4ae452127
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:24:22 2025 +0300

    «Added h1 tag»

commit cd073bfa2b66136c94b6ee7d8ce1a57a89fceff5
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:20:57 2025 +0300

    Initial Commit

```

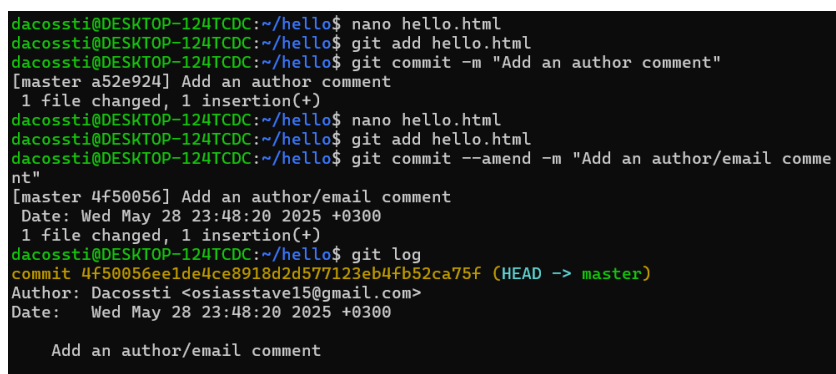
Рис. 4.13: Удаление тега оорс

## 4.12 Изменение предыдущего коммита

Добавим в страницу комментарий автора.

```
<!-- Author: Dmitry S. Kulyabov -->
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

Затем добавим их в репозиторий. Теперь мы хотим добавить в комментарий автора почту, обнови страницу hello, включив в неё почту. Чтобы у нас остался один коммит, а не два, изменим последний с помощью опции `--amend`, теперь в логах отображается последняя версия коммита (рис. 4.14).



```
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git add hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git commit -m "Add an author comment"
[master a52e924] Add an author comment
1 file changed, 1 insertion(+)
dacossti@DESKTOP-124TCDC:~/hello$ nano hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git add hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git commit --amend -m "Add an author/email comment"
[master 4f50056] Add an author/email comment
Date: Wed May 28 23:48:20 2025 +0300
1 file changed, 1 insertion(+)
dacossti@DESKTOP-124TCDC:~/hello$ git log
commit 4f50056ee1de4ce8918d2d577123eb4fb52ca75f (HEAD -> master)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:48:20 2025 +0300

    Add an author/email comment
```

Рис. 4.14: Изменение предыдущего коммита

## 4.13 Перемещение файлов

Переместим наш файл в каталог `lib`. Для этого создадим его и используем команду `git mv`, сделаем коммит этого перемещения (рис. 4.15).

```
dacossti@DESKTOP-124TDCD:~/hello$ mkdir lib
dacossti@DESKTOP-124TDCD:~/hello$ git mv hello.html lib
dacossti@DESKTOP-124TDCD:~/hello$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    renamed:   hello.html -> lib/hello.html

dacossti@DESKTOP-124TDCD:~/hello$ mkdir lib
mkdir: cannot create directory 'lib': File exists
dacossti@DESKTOP-124TDCD:~/hello$ mv hello.html lib
mv: cannot stat 'hello.html': No such file or directory
dacossti@DESKTOP-124TDCD:~/hello$ git commit -m "Moved hello.html to lib"
[master a11b2a8] Moved hello.html to lib
1 file changed, 0 insertions(+), 0 deletions(-)
rename hello.html => lib/hello.html (100%)
```

Рис. 4.15: Перемещение файлов

## 4.14 Подробнее о структуре

Добавим файл index.html в наш репозиторий

```
<html>
  <body>
    <iframe src="lib/hello.html" width="200" height="200" />
  </body>
</html>
```

Добавим файл и сделаем коммит(рис. 4.16).

```
dacossti@DESKTOP-124TDCD:~/hello$ nano index.html
dacossti@DESKTOP-124TDCD:~/hello$ git add index.html
dacossti@DESKTOP-124TDCD:~/hello$ git commit -m "Added index.html."
[master f3fe4cf] Added index.html.
1 file changed, 5 insertions(+)
create mode 100644 index.html
dacossti@DESKTOP-124TDCD:~/hello$ cat index.html
<html>
<body>
<iframe src="lib/hello.html" width="200" height="200" />
</body>
</html>
```

Рис. 4.16: Добавление нового файла в репозиторий

Теперь при открытии index.html, увидим кусок страницы hello в маленьком окошке(рис. 4.17).

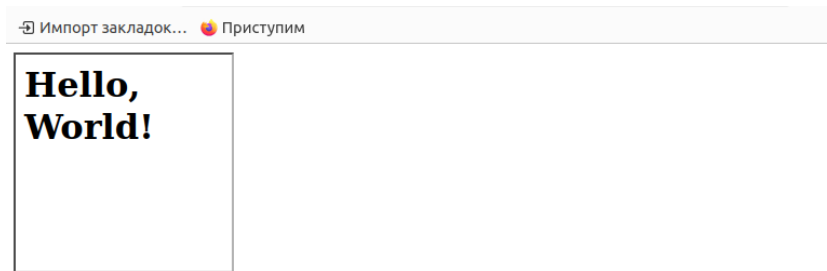


Рис. 4.17: Добавление нового файла в репозиторий

## 4.15 Git внутри: Каталог .git

Просмотрим каталог, в котором хранится вся информация git. Затем посмотрим набор каталогов, имена которых состоят из 2 символов. Имена каталогов являются первыми двумя буквами хэша sha1 объекта, хранящегося в git. Посмотрим в один из каталогов с именем из 2 букв. Увидим файлы с именами из 38 символов. Это файлы, содержащие объекты, хранящиеся в git. Посмотрим файл конфигурации, создающийся для каждого конкретного проекта. Затем посмотрим подкаталоги `.git/refs/heads` и `.git/refs/tags`, а также содержимое файла `v1`, в нём хранится хэш коммита, привязанный к тегу. Также посмотрим содержимое файла `HEAD`, который содержит ссылку на текущую ветку, в данный момент это ветка `master`(рис. 4.18).

```

dacossti@DESKTOP-124TDCD:~/hello$ ls -C .git
branches      config      HEAD      index      logs      ORIG_HEAD  refs
COMMIT_EDITMSG description hooks      info      objects    packed-refs
dacossti@DESKTOP-124TDCD:~/hello$ ls -C .git/objects
00 1a 3c 52 62 84 8a a1 b5 bd cd f3 f7 fa pack
09 26 4f 56 68 88 91 a5 b6 cb f0 f4 f8 info
dacossti@DESKTOP-124TDCD:~/hello$ ls -C .git/objects/1a
44782cac245a2654d36f7f63a06a45618bf504
dacossti@DESKTOP-124TDCD:~/hello$ cat .git/config
[core]
    repositoryformatversion = 0
    filemode = true
    bare = false
    logallrefupdates = true
dacossti@DESKTOP-124TDCD:~/hello$ ls .git/refs
heads tags
dacossti@DESKTOP-124TDCD:~/hello$ ls .git/refs/heads
master
dacossti@DESKTOP-124TDCD:~/hello$ ls .git/refs/tags
v1 v1-beta
dacossti@DESKTOP-124TDCD:~/hello$ cat .git/refs/tags/v1
1a44782cac245a2654d36f7f63a06a45618bf504
dacossti@DESKTOP-124TDCD:~/hello$ cat .git/HEAD
ref: refs/heads/master

```

Рис. 4.18: Каталог .git

## 4.16 Работа непосредственно с объектами git

Найдем последний коммит и выведем его с помощью SHA1 хэша. Затем посмотрим дерево каталогов, ссылка на который идёт в последнем коммите, выведем каталог lib и файл hello.html(рис. 4.19).

```

dacossti@DESKTOP-124TDCD:~/hello$ git log --max-count=1
commit f3fe4cf7f989f1ed070c12642beb496e18e2a2c0 (HEAD -> master)
Author: Dacossti <osiasstave15@gmail.com>
Date:   Wed May 28 23:53:39 2025 +0300

    Added index.html.
dacossti@DESKTOP-124TDCD:~/hello$ git cat-file -t f3fe4cf
commit
dacossti@DESKTOP-124TDCD:~/hello$ git cat-file -p f3fe4cf
tree f81012e7305fef6023972bcd54f999d24c80339b
parent a11b2a803e8ebc6d86c03534793802779c0fa627
author Dacossti <osiasstave15@gmail.com> 1748465619 +0300
committer Dacossti <osiasstave15@gmail.com> 1748465619 +0300

Added index.html.
dacossti@DESKTOP-124TDCD:~/hello$ git cat-file -p f3fe4cf
tree f81012e7305fef6023972bcd54f999d24c80339b
parent a11b2a803e8ebc6d86c03534793802779c0fa627
author Dacossti <osiasstave15@gmail.com> 1748465619 +0300
committer Dacossti <osiasstave15@gmail.com> 1748465619 +0300

Added index.html.

```

Рис. 4.19: Работа непосредственно с объектами git

Исследуем git репозиторий вручную самостоятельно. Используя хэш родительского коммита последовательно дойдем до первой версии файла hello.html и посмотрим его(рис. 4.20).

```
dacossti@DESKTOP-124TDCD:~/hello$ git cat-file -p f81012e7305fef6023972bcd54f999d2
4c80339b
100644 blob 68583eacd328506a31568f0df6f45ebb874238b5    index.html
040000 tree 5619267058d80fdce15b5177f5425be1dc5185ca    lib
dacossti@DESKTOP-124TDCD:~/hello$ git cat-file -p 5619267058d80fdce15b5177f5425be1dc5185ca
100644 blob fa1eedcd4549903a20cb27570a976941a6155d41    hello.html
dacossti@DESKTOP-124TDCD:~/hello$ git cat-file -p fa1eedcd4549903a20cb27570a976941a6155d41
<!-- Author: Osias Stave I. D. (osiasstave15@gmail.com)-->
<html>
<head>
</head>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

Рис. 4.20: Поиск оригинального файла hello.html

## 4.17 Создание ветки

Создадим новую ветку «style» и перейдем в неё. Добавим туда файл стилей style.css и добавим его в репозиторий. Обновим файл hello.html, чтобы использовать стили style.css и index.html, также обавим их в репозиторий(рис. 4.21).

```
dacossti@DESKTOP-124TDCD:~/hello$ git checkout -b style
Switched to a new branch 'style'
dacossti@DESKTOP-124TDCD:~/hello$ git status
On branch style
nothing to commit, working tree clean
dacossti@DESKTOP-124TDCD:~/hello$ touch lib/style.css
dacossti@DESKTOP-124TDCD:~/hello$ nano lib/style.css
dacossti@DESKTOP-124TDCD:~/hello$ git add lib/style.css
dacossti@DESKTOP-124TDCD:~/hello$ git commit -m "Added css stylesheet"
[style 35235b4] Added css stylesheet
1 file changed, 3 insertions(+)
create mode 100644 lib/style.css
dacossti@DESKTOP-124TDCD:~/hello$ nano lib/hello.html
dacossti@DESKTOP-124TDCD:~/hello$ nano lib/hello.html
dacossti@DESKTOP-124TDCD:~/hello$ git add lib/hello.html
dacossti@DESKTOP-124TDCD:~/hello$ git commit -m "Hello uses style.css"
[style 51d1d57] Hello uses style.css
1 file changed, 2 insertions(+)
```

Рис. 4.21: Создание ветки



## 4.18 Навигация по веткам

Посмотрим все логи(рис. 4.22).

```
dacossti@DESKTOP-124TDCD:~/hello$ git log --all
commit 51d1d572df3ff85d4f6f039b9105044dcdbc0adf (HEAD -> style)
Author: Dacossti <osiasstave15@gmail.com>
Date: Thu May 29 00:05:37 2025 +0300

    Hello uses style.css

commit 35235b4e9968b7418781640e6cc44c27b0812f58
Author: Dacossti <osiasstave15@gmail.com>
Date: Thu May 29 00:03:53 2025 +0300

    Added css stylesheet

commit f3fe4cf7f989f1ed070c12642beb496e18e2a2c0 (master)
Author: Dacossti <osiasstave15@gmail.com>
Date: Wed May 28 23:53:39 2025 +0300

    Added index.html.

commit a11b2a803e8ebc6d86c03534793802779c0fa627
Author: Dacossti <osiasstave15@gmail.com>
Date: Wed May 28 23:52:42 2025 +0300

    Moved hello.html to lib

commit 4f50056ee1de4ce8918d2d577123eb4fb52ca75f
Author: Dacossti <osiasstave15@gmail.com>
Date: Wed May 28 23:48:20 2025 +0300

    Add an author/email comment
```

Рис. 4.22: Просмотр логов новой ветки

Переключимся обратно на основную ветку и посмотрим содержимое файла `lib/hello.html`, заметим, что он не использует стили, также посмотрим содержимое этого файла в новой ветке(рис. 4.23).

```
dacossti@DESKTOP-124TDCD:~/hello$ git checkout master
error: Your local changes to the following files would be overwritten by checkout:
lib/hello.html
Please commit your changes or stash them before you switch branches.
Aborting
dacossti@DESKTOP-124TDCD:~/hello$ cat lib/hello.html
<!-- Author: Osias Stave I. D. (osiasstave15@gmail.com)-->
<html>
<head>
<link type="text/css" rel="stylesheet"
media="all" href="lib/style.css" />
</head>
<body>
<iframe src="lib/hello.html" width="200" height="200" />
</body>
</html>
```

Рис. 4.23: Переключение между ветками

## 4.19 Изменения в ветке master

Вернемся в основную ветку и добавим файл README.md. Просмотрим ветки и их различия(рис. 4.24, 4.25).

```
dacossti@DESKTOP-124TCDC:~/hello$ touch README.md
dacossti@DESKTOP-124TCDC:~/hello$ echo "This is the Hello World example from the g
it tutorial." > README.md
dacossti@DESKTOP-124TCDC:~/hello$ git add README.md
dacossti@DESKTOP-124TCDC:~/hello$ git commit -m "Added README"
[style c8868ca] Added README
1 file changed, 1 insertion(+)
create mode 100644 README.md
```

Рис. 4.24: Изменения в ветке master

```
dacossti@DESKTOP-124TCDC:~/hello$ git log --graph --all
* commit c8868ca70001051bf0c5f7ca4c2b133c117f92 (HEAD -> style)
 | Author: Dacossti <osiasstave15@gmail.com>
 | Date: Thu May 29 00:11:23 2025 +0300
 |
 | Added README
 |
* commit 51d1d572df3ff85d4f6f039b9105044dcdcb0adf
 | Author: Dacossti <osiasstave15@gmail.com>
 | Date: Thu May 29 00:05:37 2025 +0300
```

Рис. 4.25: Просмотр веток

## 4.20 Слияние

Слияние переносит изменения из двух веток в одну. Вернемся к ветке style и сольем master с style(рис. 4.26).

```
dacossti@DESKTOP-124TCDC:~/hello$ git checkout style
M lib/hello.html
Already on 'style'
dacossti@DESKTOP-124TCDC:~/hello$ git merge master
Already up to date.
dacossti@DESKTOP-124TCDC:~/hello$ git log --graph --all
```

Рис. 4.26: Слияние веток

## 4.21 Создание конфликта

Вернемся в ветку master и создадим конфликт, внося изменения в файл hello.html. Просмотрим ветки. После коммита «Added README» ветка master была объединена с веткой style, но в настоящее время в master есть дополнительный коммит, который не был слит с style. Последнее изменение в master конфликтует с некоторыми изменениями в style(рис. 4.27).

```
dacossti@DESKTOP-124TCDC:~/hello$ nano lib/hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git add lib/hello.html
dacossti@DESKTOP-124TCDC:~/hello$ git commit -m 'Life is great'
```

Рис. 4.27: Создание конфликта

## 4.22 Разрешение конфликтов

Вернемся к ветке style и попытаемся объединить ее с новой веткой master. В файле lib/hello.html можно увидеть записи с обеих версий этого файла. Первый раздел — версия текущей ветки (style). Второй раздел — версия ветки master. Внесем изменения в lib/hello.html, оставив только необходимую нам запись и добавим этот файл в репозиторий, чтобы вручную разрешить конфликт(рис. 4.28).

```
dacossti@DESKTOP-124TCDC:~/hello$ git checkout style
Already on 'style'
dacossti@DESKTOP-124TCDC:~/hello$ git merge master
Already up to date.
dacossti@DESKTOP-124TCDC:~/hello$ cat lib/hello.html
```

Рис. 4.28: Разрешение конфликта

## 4.23 Сброс ветки style

Вернемся на ветке style к точке перед тем, как мы слили ее с веткой master. Мы хотим вернуться в ветке style в точку перед слиянием с master. Нам необходимо найти последний коммит перед слиянием(рис. 4.29).

```
dacossti@DESKTOP-124TDCD:~/hello$ git checkout style
Already on 'style'
dacossti@DESKTOP-124TDCD:~/hello$ git log --graph
* commit 8148247a7246454dc09196039cd47edfa85bdb95 (HEAD -> style)
| Author: Dacossti <osiasstave15@gmail.com>
| Date: Thu May 29 00:20:36 2025 +0300
|
```

Рис. 4.29: Поиск коммита перед слиянием

Мы видим, что коммит «Updated index.html» был последним на ветке style перед слиянием. Сбросим ветку style к этому коммиту(рис. 4.30).

```
dacossti@DESKTOP-124TDCD:~/hello$ git reset --hard 8148247a7246454dc09196039cd47edfa85bdb95
HEAD is now at 8148247 Updated index.html
```

Рис. 4.30: Сброс ветки style

Поищем лог ветки style. Увидим, что у нас в истории больше нет коммитов слияний.

## 4.24 Сброс ветки master

Добавив интерактивный режим в ветку master, мы внесли изменения, конфликтующие с изменениями в ветке style. Давайте вернемся в ветку master в точку перед внесением конфликтующих изменений. Это позволяет нам продемонстрировать работу команды git rebase, не беспокоясь о конфликтах. Просмотрим коммиты ветки master(рис. 4.31).

```
dacossti@DESKTOP-124TDCD:~/hello$ git checkout master
Switched to branch 'master'
dacossti@DESKTOP-124TDCD:~/hello$ git log --graph
* commit f3fe4cf7f989f1ed070c12642beb496e18e2a2c0 (HEAD -> master)
| Author: Dacossti <osiasstave15@gmail.com>
| Date: Wed May 28 23:53:39 2025 +0300
|
```

Рис. 4.31: Поиск коммита перед конфликтом

Коммит «Added README» идет непосредственно перед коммитом конфликтующего интерактивного режима. Мы сбросим ветку master к коммиту «Added README»(рис. 4.32).

```
dacossti@DESKTOP-124TCDC:~/hello$ git reset --hard ecf2b8c3fa595d43c1f84031ebde5b97173878de
HEAD is now at ecf2b8c Added README
```

Рис. 4.32: Сброс ветки master

## 4.25 Перебазирование

Используем команду rebase вместо команды merge. Мы вернулись в точку до первого слияния и хотим перенести изменения из ветки master в нашу ветку style. На этот раз для переноса изменений из ветки master мы будем использовать команду git rebase вместо слияния(рис. 4.33).

```
dacossti@DESKTOP-124TCDC:~/hello$ git checkout style
Switched to branch 'style'
dacossti@DESKTOP-124TCDC:~/hello$ git rebase master
```

Рис. 4.33: Перебазирование

## 4.26 Слияние в ветку master

Вернемся в ветку master и сольем ветку style в неё с помощью команды git merge(рис. 4.34).

```
dacossti@DESKTOP-124TCDC:~/hello$ git checkout master
Switched to branch 'master'
dacossti@DESKTOP-124TCDC:~/hello$ git merge style
Updating ecf2b8c..6fab7f4
Fast-forward
 lib/hello.html | 4 +++-
 lib/style.css   | 3 +++
 2 files changed, 6 insertions(+), 1 deletion(-)
 create mode 100644 lib/style.css
dacossti@DESKTOP-124TCDC:~/hello$ git log
commit 6fab7f4f8057362ffbf8ae567f78c0f6723b29ed (HEAD -> master, style)
Author: Dacossti <osiasstave15@gmail.com>
Date: Thu May 29 00:20:36 2025 +0300

    Updated index.html
```

Рис. 4.34: Слияние style в master

## 4.27 Клонирование репозитория

Перейдем в наш рабочий каталог и сделаем клон репозитория hello, затем создадим клон репозитория. Просмотрев его увидим список всех файлов на верхнем уровне оригинального репозитория README.md, index.html и lib. Затем просмотрим историю репозитория и увидим список всех коммитов в новый репозиторий, и он совпадает с историей коммитов в оригинальном репозитории. Единствен в названиях веток(рис. 4.35).

```
dacossti@DESKTOP-124TCDC:~/hello$ cd ..
dacossti@DESKTOP-124TCDC:~$ pwd
/home/dacossti
dacossti@DESKTOP-124TCDC:~$ ls
frames_2025-04-25_04.30.34.gv  hello          ros2_backup
frames_2025-04-25_04.30.34.pdf llm_ros2_humble test.py
dacossti@DESKTOP-124TCDC:~$ git clone hello cloned_hello
Cloning into 'cloned_hello'...
done.
dacossti@DESKTOP-124TCDC:~$ ls
cloned_hello  hello          test.py
frames_2025-04-25_04.30.34.gv llm_ros2_humble
frames_2025-04-25_04.30.34.pdf ros2_backup
dacossti@DESKTOP-124TCDC:~$ cd cloned_hello
ls
index.html  lib  README.md
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git log --all
commit 6fab7f4f8057362ffbfcae567f78c0f6723b29ed (HEAD -> master, origin/style, origin/master, origin/HEAD)
Author: Dacossti <osiasstave15@gmail.com>
Date: Thu May 29 00:20:36 2025 +0300

Updated index.html
```

Рис. 4.35: Клонирование репозитория

## 4.28 Что такое origin?

Клонированный репозиторий знает об имени по умолчанию удаленного репозитория. Посмотрим, подробную информацию об имени по умолчанию. Для того, чтобы увидеть все ветки используем опцию -a(рис. 4.36).

```
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git remote
origin
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git remote show origin
* remote origin
  Fetch URL: /home/dacossti/hello
  Push URL: /home/dacossti/hello
  HEAD branch: master
  Remote branches:
    master tracked
    style tracked
  Local branch configured for 'git pull':
    master merges with remote master
  Local ref configured for 'git push':
    master pushes to master (up to date)
```

Рис. 4.36: Просмотр имени по умолчанию удаленного репозитория

## 4.29 Удаленные ветки

Посмотрим на ветки, доступные в нашем клонированном репозитории. Можно увидеть, что в списке только ветка master(рис. 4.37).

```
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git branch
* master
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/master
  remotes/origin/style
```

Рис. 4.37: Просмотр доступных веток

## 4.30 Изменение оригинального репозитория

Перейдем в репозиторий hello. Внесем изменения в файл README.md. Затем добавим их в репозиторий(рис. 4.38).

```
dacossti@DESKTOP-124TDCD:~/cloned_hello$ cd ../hello
dacossti@DESKTOP-124TDCD:~/hello$ nano README.md
dacossti@DESKTOP-124TDCD:~/hello$ git add README.md
dacossti@DESKTOP-124TDCD:~/hello$ git commit -m "Changed README in original repo"
On branch master
nothing to commit, working tree clean
```

Рис. 4.38: Изменение оригинального репозитория

Перейдём в клон репозитория и используем команду `git fetch`, которая будет извлекать новые коммиты из удаленного репозитория, но не будет сливать их с наработками в локальных ветках(рис. 4.39).

```
dacossti@DESKTOP-124TCDC:~/hello$ cd ../cloned_hello
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git fetch
```

Рис. 4.39: Извлечение изменений

## 4.31 Слияние извлеченных изменений

Сольём внесённые изменения в главную ветку. Также можно было бы использовать команду `git pull`, которая является объединением `fetch` и `merge` в одну команду.

(рис. 4.40).

```
dacossti@DESKTOP-124TCDC:~/cloned_hello$ cat README.md
This is the Hello World example from the git tutorial.
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git merge origin/master
Already up to date.
dacossti@DESKTOP-124TCDC:~/cloned_hello$ cat README.md
This is the Hello World example from the git tutorial.
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git pull
Already up to date.
```

Рис. 4.40: Слияние извлеченных изменений

## 4.32 Добавление ветки наблюдения

Добавим локальную ветку, которая отслеживает удаленную ветку, теперь мы можем видеть ветку `style` в списке веток и логе(рис. 4.41).



```
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git branch --track style origin/style
Branch 'style' set up to track remote branch 'style' from 'origin'.
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git branch -a
* master
  style
remotes/origin/HEAD -> origin/master
remotes/origin/master
remotes/origin/style
dacossti@DESKTOP-124TCDC:~/cloned_hello$ git log --max-count=2
commit 6fab7f4f8057362ffbf8ae567f78c0f6723b29ed (HEAD -> master, origin/style, ori
gin/master, origin/HEAD, style)
Author: Dacossti <osiasstave15@gmail.com>
```

Рис. 4.41: Добавление ветки наблюдения

### 4.33 Создание чистого репозитория

Как правило, репозитории, оканчивающиеся на .git являются чистыми репозиториями. Создадим такой в рабочем каталоге. Затем добавим репозиторий hello.git к нашему оригинальному репозиторию(рис. 4.42).

```
dacossti@DESKTOP-124TCDC:~/cloned_hello$ cd ..
dacossti@DESKTOP-124TCDC:~$ git clone --bare hello hello.git
Cloning into bare repository 'hello.git'...
done.
dacossti@DESKTOP-124TCDC:~$ ls hello.git
branches  config  description  HEAD  hooks  info  objects  packed-refs  refs
dacossti@DESKTOP-124TCDC:~$ cd hello
dacossti@DESKTOP-124TCDC:~/hello$ git remote add shared ../hello.git
```

Рис. 4.42: Создание чистого репозитория

### 4.34 Отправка и извлечение изменений

Так как чистые репозитории, как правило, расшариваются на каком-нибудь сетевом сервере, нам необходимо отправить наши изменения в другие репозитории. Начнем с создания изменения для отправки. Отредактируем файл README.md и сделаем коммит, затем отправим изменения в общий репозиторий. Затем извлечем изменения из общего репозитория(рис. 4.43).

```

dacossti@DESKTOP-124TDCD:~/hello$ git checkout master
M
 README.md
Already on 'master'
dacossti@DESKTOP-124TDCD:~/hello$ git add README.md
dacossti@DESKTOP-124TDCD:~/hello$ git commit -m "Added shared comment to readme"
[master ca801ee] Added shared comment to readme
1 file changed, 1 insertion(+)
dacossti@DESKTOP-124TDCD:~/hello$ git push shared master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 404 bytes | 404.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To ../hello.git
6fab7f4..ca801ee master -> master
dacossti@DESKTOP-124TDCD:~/hello$ cd ../cloned_hello
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git remote add shared ../hello.git
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git branch --track shared master
Branch 'shared' set up to track local branch 'master'.
dacossti@DESKTOP-124TDCD:~/cloned_hello$ git pull shared master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 384 bytes | 96.00 KiB/s, done.
From ../hello

```

Рис. 4.43: Отправка и извлечение изменений

## 5 Выводы

В результате выполнения лабораторной работы были приобретены практические навыки работы с системой управления версиями Git.

## Список литературы

1. Git [Электронный ресурс]. Wikimedia Foundation, Inc., 2024. URL: <https://ru.wikipedia.org/wiki/Git>.
2. GitHub [Электронный ресурс]. GitHub, Inc., 2024. URL: <https://github.com/>.