

Лабораторная работа №1

Работа с git

Поздняков Данила Романович

Цель работы

Научиться работать с системой контроля версий Git.

Выполнение лабораторной работы

1.1 Подготовка

1.1.1 Установка имени и электронной почты

Если вы никогда ранее не использовали git, для начала вам необходимо осуществить установку. Выполните следующие команды, чтобы git узнал ваше имя и электронную почту. Если git уже установлен, можете переходить к разделу окончания строк.

```
git config --global user.name "Your Name"
```

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

1.1.2 Параметры установки окончаний строк

Настройка core.autocrlf с параметрами true и input делает все переводы строк текстовых файлов в главном репозитории одинаковы. core.autocrlf true - git автоматически конвертирует CRLF->LF при коммите и обратно LF->CRLF при выгрузке кода из репозитория на файловую систему (используют в Windows). core.autocrlf input - конвертация CRLF в LF только при коммитах (используют в Mac/Linux). Если core.safecrlf установлен в true или warn, git проверяет, если преобразование является обратимым для текущей настройки core.autocrlf. core.safecrlf true - отвержение необратимого преобразования lf<->crlf. Полезно, когда специфические бинарники похожие на текстовые файлы. core.safecrlf warn - печать только предупреждение, но принимает необратимый переход.

Для пользователей Windows:

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

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

1.1.3 Установка отображения unicode

По умолчанию, git будет печатать не-ASCII символы в именах файлов в виде восьмеричных последовательностей . Что бы избежать нечитаемых строк, установите соответствующий флаг.

```
git config --global core.quotepath off
```

1.2 Создание проекта

1.2.1 Создайте страницу «Hello, World»

Начните работу в пустом рабочем каталоге с создания пустого каталога с именем hello, затем войдите в него и создайте там файл с именем hello.html.

```
mkdir hello  
cd hello  
touch hello.html  
echo "Hello, World!" > hello.html
```

1.2.2 Создание репозитория

Чтобы создать git репозиторий из этого каталога, выполните команду git init.

```
git init
```

1.2.3 Добавление файла в репозиторий

Добавим файл в репозиторий.

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

1.2.4 Проверка состояния репозитория

Используйте команду git status, чтобы проверить текущее состояние репозитория.

```
git status
```

Команда проверки состояния сообщит, что коммитить нечего. Это означает, что в репозитории хранится текущее состояние рабочего каталога, и нет никаких изменений, ожидающих записи.

```
polya@DESKTOP-SQTC4VF MINGW64 ~
$ mkdir hello

polya@DESKTOP-SQTC4VF MINGW64 ~
$ cd hello

polya@DESKTOP-SQTC4VF MINGW64 ~/hello
$ touch hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello
$ echo "Hello, World!" > hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello
$ git init
Initialized empty Git repository in C:/Users/polya/hello/.git/
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello.html

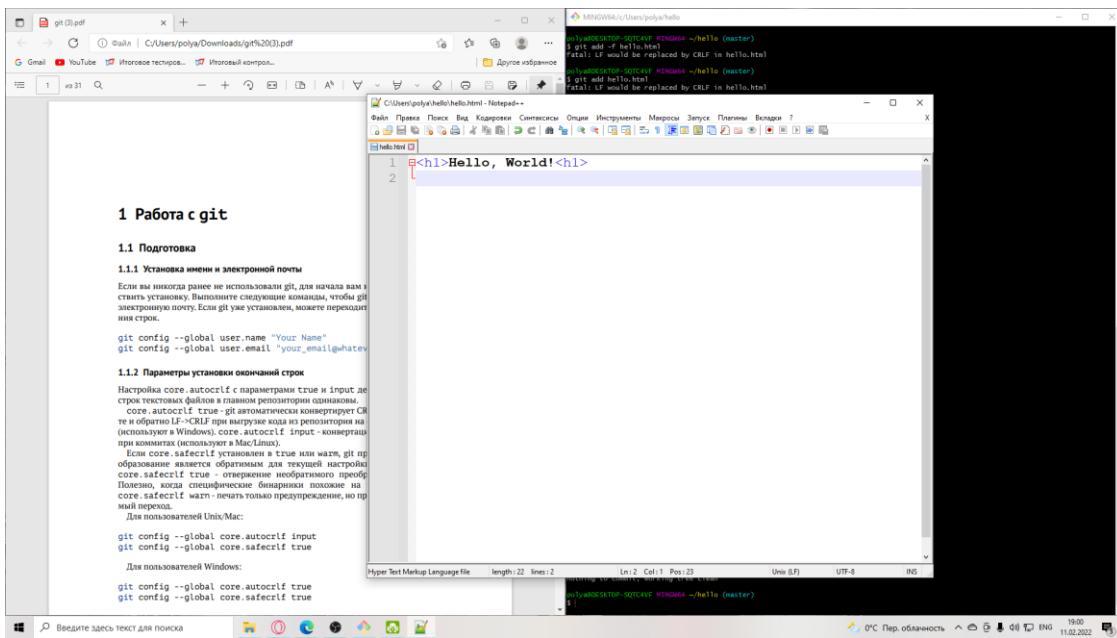
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Initial Commit 2"
[master 0028854] Initial Commit 2
 1 file changed, 1 insertion(+), 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
nothing to commit, working tree clean
```

1.3 Внесение изменений

1.3.1 Измените страницу «Hello, World»

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



Проверьте состояние рабочего каталога.

```
git status
```

git знает, что файл hello.html был изменен, но при этом эти изменения еще не зафиксированы в репозитории.

Также обратите внимание на то, что сообщение о состоянии дает вам подсказку о том, что нужно делать дальше. Если вы хотите добавить эти изменения в репозиторий, используйте команду git add. В противном случае используйте команду git checkout для отмены изменений.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified:   hello.html
```

1.4 Индексация изменений

Теперь выполните команду git, чтобы проиндексировать изменения. Проверьте состояние.

```
git add hello.html
```

```
git status
```

Изменения файла hello.html были проиндексированы. Это означает, что git теперь знает об изменениях, но изменение пока не записано в репозиторий. Следующий коммит будет включать в себя проиндексированные изменения. Если вы решили, что не хотите коммитить изменения, команда состояния напомнит вам о том, что с помощью команды git reset можно снять индексацию этих изменений. Отдельный шаг индексации в git позволяет вам продолжать вносить изменения в рабочий каталог, а

затем, в момент, когда вы захотите взаимодействовать с версионным контролем, git позволит записать изменения в малых коммитах, которые фиксируют то, что вы сделали. Разделяя индексацию и коммит, вы имеете возможность с легкостью настроить, что идет в какой коммит.

1.4.1 Коммит изменений

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 и проверьте состояние.

```
git commit
```

Откроется редактор. В первой строке введите комментарий: «Added h1 tag». Сохраните файл и выйдите из редактора (для этого в редакторе по-умолчанию (Vim) вам нужно нажать клавишу `ESC`, ввести `:wq` и нажать `Enter`). Теперь еще раз проверим состояние.

```
git status
```

Рабочий каталог чистый, можно продолжить работу.

1.4.2 Добавьте стандартные теги страницы

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

и

.

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

Проверьте текущий статус:
git status
Обратите внимание на то, что hello.html указан дважды
вое изменение (добавление стандартных тегов) проиндексировано
коммиту. Второе изменение (добавление заголовков HTML) яв-
ляется непроиндексированным. Если бы вы делали коммит сейчас, заголовки не
были бы индексированы.
Если бы вы делали коммит сейчас, заголовки не
были бы индексированы.
Произведите коммит проиндексированного изменения (заголовков), а затем еще раз проверьте состояние.
git commit -m "Added standard HTML page tags"
git status
Состояние команды говорит о том, что hello.html имеет и
изменения, но уже не в буферной зоне.
Теперь добавьте второе изменение в индекс, а затем про-
изведите коммит с помощью команды git status.
git add .
git status
В качестве файла для добавления, мы использовали текущий
крайний и удаленный путь для отслеживания всех изменений в файле
hello.html. Несмотря на то что мы добавили все, не имеющие
состава перед запуском add, просто чтобы убедиться, что мы
то файла, который добавлять было не нужно.
Второе изменение было проиндексировано и готово к коммиту.
Сделайте коммит второго изменения

```

Теперь добавьте это изменение в индекс git.

```
git add hello.html
```

Теперь добавьте заголовки HTML (секцию

) к странице «Hello, World».

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

Проверьте текущий статус:
git status
Обратите внимание на то, что hello.html указан дважды
вое изменение (добавление стандартных тегов) проиндексировано
коммиту. Второе изменение (добавление заголовков HTML) яв-
ляется непроиндексированным. Если бы вы делали коммит сейчас, заголовки не
были бы индексированы.
Если бы вы делали коммит сейчас, заголовки не
были бы индексированы.
Произведите коммит проиндексированного изменения (заголовков), а затем еще раз проверьте состояние.
git commit -m "Added standard HTML page tags"
git status
Состояние команды говорит о том, что hello.html имеет и
изменения, но уже не в буферной зоне.
Теперь добавьте второе изменение в индекс, а затем про-
изведите коммит с помощью команды git status.
git add .
git status
В качестве файла для добавления, мы использовали текущий
крайний и удаленный путь для отслеживания всех изменений в файле
hello.html. Несмотря на то что мы добавили все, не имеющие
состава перед запуском add, просто чтобы убедиться, что мы
то файла, который добавлять было не нужно.
Второе изменение было проиндексировано и готово к коммиту.
Сделайте коммит второго изменения

```

Проверьте текущий статус:

```
git status
```

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

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

```
git status
```

Состояние команды говорит о том, что hello.html имеет незафиксированные изменения, но уже не в буферной зоне. Теперь добавьте второе изменение в индекс, а затем проверьте состояние с помощью команды git status.

```
git add .
```

```
git status
```

В качестве файла для добавления, мы использовали текущий каталог (.). Это краткий и удобный путь для добавления всех изменений в файлы текущего каталога и его подкаталоги. Но поскольку он добавляет все, не лишним будет проверить состояние перед запуском add, просто чтобы убедиться, что вы не добавили какого-то файла, который добавлять было не нужно. Второе изменение было проиндексировано и готово к коммиту. Сделайте коммит второго изменения

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

1.4.3 История

Получим список произведенных изменений:

```
git log
```

Однострочный формат истории:

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

Есть много вариантов отображения лога.

```
git log --pretty=oneline --max-count=2
```

```
git log --pretty=oneline --since='5 minutes ago'
```

```
git log --pretty=oneline --until='5 minutes ago'
```

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

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

Справочная страница:

```
man git-log
```

Инструмент gitk полезен в изучении истории изменений.

1.4.4 Получение старых версий

Возвращаться назад в историю очень просто. Команда `checkout` скопирует любой снимок из репозитория в рабочий каталог. Получите хэши предыдущих версий

```
git log
```

Изучите данные лога и найдите хэш для первого коммита. Он должен быть в последней строке данных. Используйте этот хэш-код (достаточно первых 7 знаков) в команде ниже. Затем проверьте содержимое файла `hello.html`.

```
git checkout
```

```
cat hello.html
```

Вернитесь к последней версии в ветке `master`

```
git checkout master
```

```
cat hello.html
```

`master` — имя ветки по умолчанию. Переключая имена веток, вы попадаете на последнюю версию выбранной ветки.

1.4.5 Создание тегов версий

Давайте назовем текущую версию страницы `hello` первой (`v1`). Создайте тег первой версии

```
git tag v1
```

Теперь текущая версия страницы называется `v1`. Теги для предыдущих версий Давайте создадим тег для версии, которая идет перед текущей версией и назовем его `v1-beta`. В первую очередь нам надо переключиться на предыдущую версию. Вместо поиска до хэш, мы будем использовать `^`, обозначающее «родитель `v1`». Вместо обозначения `v1^` можно использовать `v1~1`. Это обозначение можно определить как «первую версию предшествующую `v1`».

```
git checkout v1^
```

```
cat hello.html
```

Это версия с тегами

и

, но еще пока без

. Давайте сделаем ее версией `v1-beta`.

```
git tag v1-beta
```

1.4.6 Переключение по имени тега

Теперь попробуйте попереключаться между двумя отмеченными версиями.

```
git checkout v1
```

```
git checkout v1-beta
```

1.4.7 Просмотр тегов с помощью команды tag

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

```
git tag
```

Вы также можете посмотреть теги в логе.

```
git log master --all
```

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

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Initial Commit 2"
[master 0028854] Initial Commit 2
 1 file changed, 1 insertion(+), 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
nothing to commit, working tree clean

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello.html

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

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Added standard HTML page tags"
[master e7eec14] Added standard HTML page tags
 1 file changed, 5 insertions(+), 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   hello.html

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

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Added standard HTML page tags"
[master 66a7b5d] Added standard HTML page tags
 1 file changed, 2 insertions(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log
commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (HEAD -> master)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

    Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline
66a7b5d4a9bbeaeee81ef1508aa3f7b67f0c0b648 (HEAD -> master) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline --max-count=2
66a7b5d4a9bbeaeee81ef1508aa3f7b67f0c0b648 (HEAD -> master) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d Added standard HTML page tags

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline --since='5 minutes ago'
66a7b5d4a9bbeaeee81ef1508aa3f7b67f0c0b648 (HEAD -> master) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d Added standard HTML page tags

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline --until='5 minutes ago'
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline --author=IvanPolyakovNPI
66a7b5d4a9bbeaeee81ef1508aa3f7b67f0c0b648 (HEAD -> master) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline --all
66a7b5d4a9bbeaeee81ef1508aa3f7b67f0c0b648 (HEAD -> master) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log
commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (HEAD -> master)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

    Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout 183cce074921
Note: switching to '183cce074921'.

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 183cce0 Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((183cce0...))
$ cat hello.html
Hello, World!

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((183cce0...))
$ git checkout master
Previous HEAD position was 183cce0 Initial Commit
Switched to branch 'master'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ cat hello.html
<html>
    <head>
    </head>
    <body>
        <h1>Hello, World!</h1>
    </body>
</html>
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git tag v1

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ 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 e7eec14 Added standard HTML page tags

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((e7eec14...))
$ cat hello.html
<html>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((e7eec14...))
$ git tag v1-beta

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((v1-beta))
$ git checkout v1
Previous HEAD position was e7eec14 Added standard HTML page tags
HEAD is now at 66a7b5d Added standard HTML page tags

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((v1))
$ git checkout v1-beta
Previous HEAD position was 66a7b5d Added standard HTML page tags
HEAD is now at e7eec14 Added standard HTML page tags

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((v1-beta))
$ git tag
v1
v1-beta

polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((v1-beta))
$ git log master --all
commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1, master)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (HEAD, tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300
```

1.5 Отмена локальных изменений (до индексации)

1.5.1 Переключитесь на ветку master

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

```
git checkout master
```

1.5.2 Измените hello.html

Иногда случается, что вы изменили файл в рабочем каталоге, и хотите отменить последние коммиты. С этим справится команда git checkout. Внесите изменение в файл hello.html в виде нежелательного комментария.

The screenshot shows a Windows desktop environment. In the center, there is a Notepad window titled 'C:\Users\poly\hello\Hello.html - Notepad+'. It contains the following HTML code:

```
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
    <!-- This is a bad comment. We want to revert it. -->
  </body>
</html>
```

Below the Notepad window is a terminal window with the title 'MINGW64 /c/Users/poly/hello'. It displays the output of the 'git status' command:

```
HEAD is now at e7ee14 Added standard HTML page tags
 1 file changed, 1 insertion(+)
 create mode 100644 hello (e7ee14...)
```

1.5.3 Проверьте состояние

Сначала проверьте состояние рабочего каталога.

```
git status
```

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

1.5.4 Отмена изменений в рабочем каталоге

Используйте команду git checkout для переключения версии файла hello.html в репозитории.

```
git checkout hello.html
```

```
git status
```

```
cat hello.html
```

Команда git status показывает нам, что не было произведено никаких изменений, не зафиксированных в рабочем каталоге.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello ((v1-beta))
$ git checkout master
Previous HEAD position was e7eec14 Added standard HTML page tags
Switched to branch 'master'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified:   hello.html

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

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout hello.html
Updated 1 path from the index

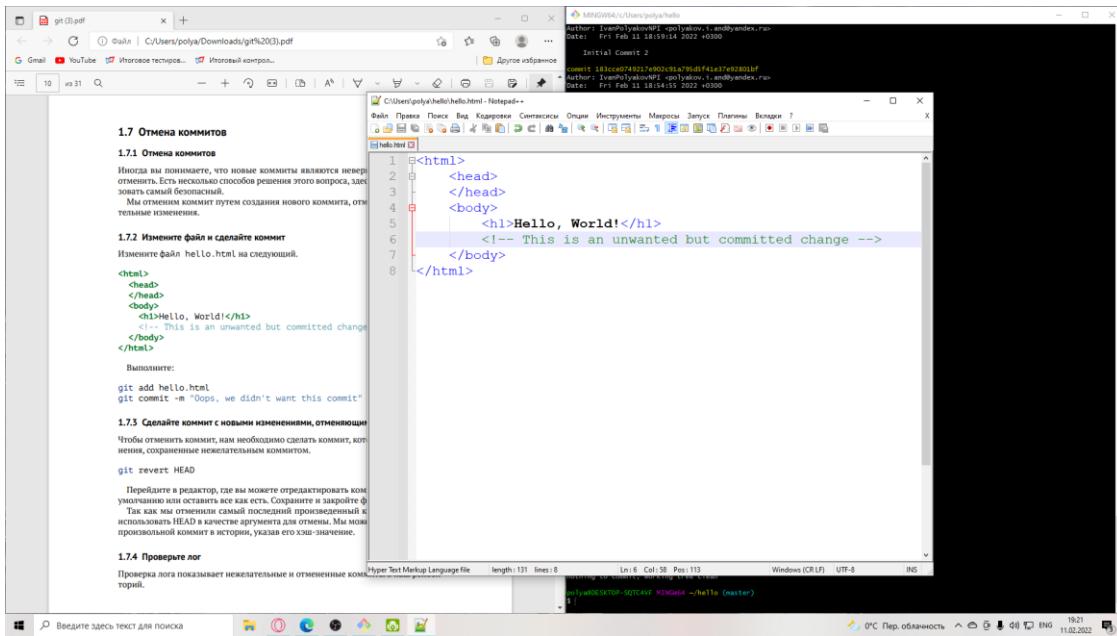
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
nothing to commit, working tree clean

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ cat hello.html
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

1.6 Отмена проиндексированных изменений (перед коммитом)

1.6.1 Измените файл и проиндексируйте изменения

Внесите изменение в файл hello.html в виде нежелательного комментария



Проиндексируйте это изменение.

```
git add hello.html
```

1.6.2 Проверьте состояние

Проверьте состояние нежелательного изменения.

```
git status
```

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

1.6.3 Выполните сброс буферной зоны

К счастью, вывод состояния показывает нам именно то, что мы должны сделать для отмены индексации изменения.

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

Команда git reset сбрасывает буферную зону к HEAD. Это очищает буферную зону от изменений, которые мы только что проиндексировали. Команда git reset (по умолчанию) не изменяет рабочий каталог. Поэтому рабочий каталог все еще содержит нежелательный комментарий. Мы можем использовать команду git checkout, чтобы удалить нежелательные изменения в рабочем каталоге. ### 1.6.4 Переключитесь на версию коммита

```
git checkout hello.html
```

```
git status
```

Наш рабочий каталог опять чист.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git reset HEAD hello.html
Unstaged changes after reset:
M       hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout hello.html
Updated 1 path from the index

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
nothing to commit, working tree clean

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ |
```

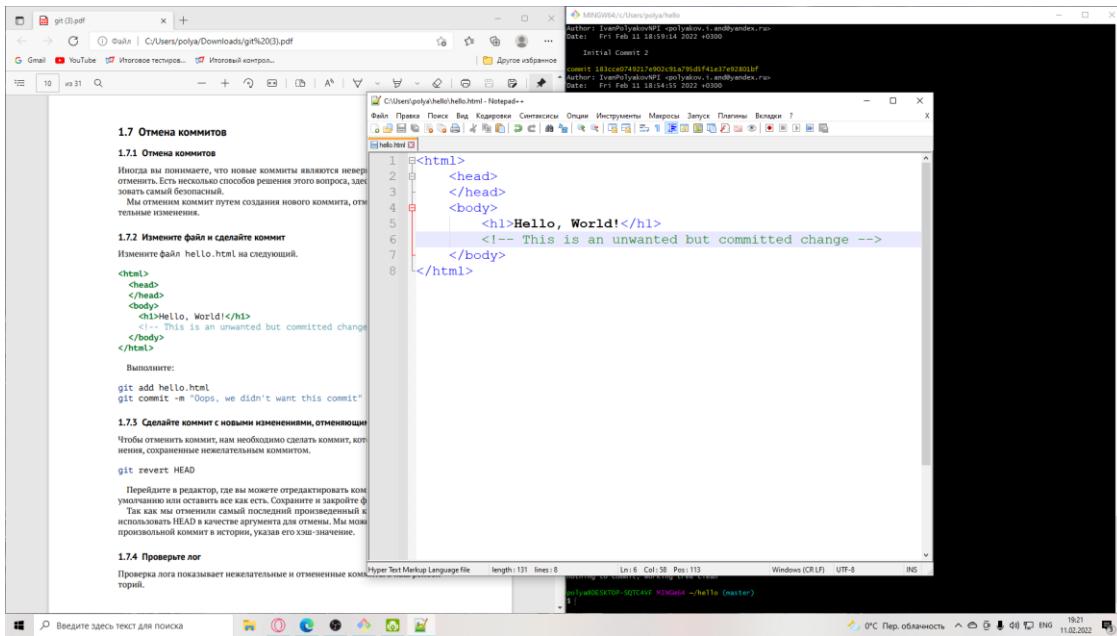
1.7 Отмена коммитов

1.7.1 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.

1.7.2 Измените файл и сделайте коммит

Измените файл hello.html на следующий.



Выполните:

```
git add hello.html
```

```
git commit -m "Oops, we didn't want this commit"
```

1.7.3 Сделайте коммит с новыми изменениями, отменяющими предыдущие

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. Сохраните и закройте файл. Так как мы отменили самый последний произведенный коммит, мы смогли использовать HEAD в качестве аргумента для отмены. Мы можем отменить любой произвольной коммит в истории, указав его хэш-значение.

1.7.4 Проверьте лог

Проверка лога показывает нежелательные и отмененные коммиты в наш репозиторий.

```
git log
```

Эта техника будет работать с любым коммитом.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Oops, we didn't want this commit"
[master 3cea0f2] Oops, we didn't want this commit
 1 file changed, 1 insertion(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git revert HEAD
[master 52dd2e0] Revert "Oops, we didn't want this commit"
 1 file changed, 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log
commit 52dd2e0aa0ad0e2f0277e97e5d33c7d823d949d8 (HEAD -> master)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:22:27 2022 +0300

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

    This reverts commit 3cea0f2fe95be8c8e331477edfa5f1c60b766ad3.

commit 3cea0f2fe95be8c8e331477edfa5f1c60b766ad3
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:22:11 2022 +0300

    Oops, we didn't want this commit

commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags
```

1.8 Удаление коммитов из ветки

git revert является мощной командой, которая позволяет отменить любые коммиты в репозиторий. Однако, и оригинальный и «отмененный» коммиты видны в истории ветки (при использовании команды git log). Часто мы делаем коммит, и сразу понимаем, что это была ошибка. Было бы неплохо иметь команду «возврата», которая позволила бы нам сделать вид, что неправильного коммита никогда и не было. Команда «возврата» даже предотвратила бы появление нежелательного коммита в истории git log.

1.8.1 Команда git reset

При получении ссылки на коммит (т.е. хэш, ветка или имя тега), команда git reset:

- перепишет текущую ветку, чтобы она указывала на нужный коммит;
- опционально сбросит буферную зону для соответствия с указанным коммитом;
- опционально сбросит рабочий каталог для соответствия с указанным коммитом.

1.8.2 Проверьте нашу историю

Давайте сделаем быструю проверку нашей истории коммитов. Выполните:

```
git log
```

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

1.8.3 Для начала отметьте эту ветку

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

```
git tag oops
```

1.8.4 Сброс коммитов к предшествующим коммиту Oops

Глядя на историю лога, 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 (если она не имеет тега, мы можем использовать хэш-значение).

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

```
git log
```

Наша ветка master теперь указывает на коммит v1, а коммитов Oops и Revert Oops в ветке уже нет. Параметр `--hard` указывает, что рабочий каталог должен быть обновлен в соответствии с новым `head` ветки.

1.8.5 Ничего никогда не теряется

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
```

Мы видим, что ошибочные коммиты не исчезли. Они все еще находятся в репозитории. Просто они отсутствуют в ветке `master`. Если бы мы не отметили их тегами, они по-прежнему находились бы в репозитории, но не было бы никакой возможности ссылаться на них, кроме как при помощи их хэш имен. Коммиты, на которые нет ссылок, остаются в репозитории до тех пор, пока не будет запущен сборщик мусора.

1.8.6 Опасность сброса

Сброс в локальных ветках, как правило, безопасен. Последствия любой «аварии» как правило, можно восстановить простым сбросом с помощью нужного коммита. Однако, если ветка «расширена» на удаленных репозиториях, сброс может сбить с толку других пользователей ветки.

```
Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git tag "oops"

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git reset --hard v1
HEAD is now at 66a7b5d Added standard HTML page tags

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline
66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (HEAD -> master, tag: v1) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta) Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --all
commit 52dd2e0aa0ad0e2f0277e97e5d33c7d823d949d8 (tag: oops)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:22:27 2022 +0300

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

    This reverts commit 3cea0f2fe95be8c8e331477edfa5f1c60b766ad3.

commit 3cea0f2fe95be8c8e331477edfa5f1c60b766ad3
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:22:11 2022 +0300

    Oops, we didn't want this commit

commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (HEAD -> master, tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

    Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$
```

1.9 Удаление тега oops

1.9.1 Удаление тега oops

Тег oops свою функцию выполнил. Давайте удалим его и коммиты, на которые он ссылался, сборщиком мусора.

```
git tag -d oops
```

```
git log -all
```

Тег «oops» больше не будет отображаться в репозитории.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git tag -d oops
Deleted tag 'oops' (was 52dd2e0)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --all
commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (HEAD -> master, tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

    Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$
```

1.10 Внесение изменений в коммиты

1.10.1 Измените страницу, а затем сделайте коммит

Добавьте в страницу комментарий автора (вставьте свою фамилию).

```
git add hello.html  
git commit --amend -m "Add an author/email comment"
```

Выполните:

```
git add hello.html
```

```
git commit -m "Add an author comment"
```

1.10.2 Необходим email

После совершения коммита вы понимаете, что любой хороший комментарий должен включать электронную почту автора. Обновите страницу hello, включив в нее email.

```
git add hello.html  
git commit --amend -m "Add an author/email comment"
```

1.10.3 Измените предыдущий коммит

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

```
git add hello.html
```

```
git commit -amend -m "Add an author/email comment"
```

1.10.4 Просмотр истории

Выполните:

```
git log
```

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

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Add an author comment"
[master 08f66d1] Add an author comment
 1 file changed, 1 insertion(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit --amend -m "Add an author comment"
[master 86708b0] Add an author comment
  Date: Fri Feb 11 19:29:44 2022 +0300
 1 file changed, 1 insertion(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline
86708b07ef213dc565c0afcb15494aea970c3491 (HEAD -> master) Add an author comment
66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta) Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit --amend -m "Add an author/email comment"
[master 50157fd] Add an author/email comment
  Date: Fri Feb 11 19:29:44 2022 +0300
 1 file changed, 1 insertion(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --pretty=oneline
50157fdf2aacd527e47adb21688faafb813a46b0 (HEAD -> master) Add an author/email comment
66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta) Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit
```

1.11 Перемещение файлов

1.11.1 Переместите файл hello.html в каталог lib

Сейчас мы собираемся создать структуру нашего репозитория. Давайте перенесем страницу в каталог lib. mkdir lib

```
git mv hello.html lib
```

```
git status
```

Перемещая файлы с помощью git mv, мы информируем git о 2 вещах:

- Что файл hello.html был удален.
- Что файл lib/hello.html был создан.
- Оба эти факта сразу же проиндексированы и готовы к коммиту. Команда git status сообщает, что файл был перемещен.

1.12 Второй способ перемещения файлов

Положительной чертой git является то, что вы можете забыть о версионном контроле до того момента, когда вы готовы приступить к коммиту кода. Что бы случилось, если бы мы использовали командную строку операционной системы для перемещения файлов вместо команды git? Следующий набор команд идентичен нашим последним действиям. Работы здесь побольше, но результат тот же. Мы могли бы выполнить:

```
mkdir lib
```

```
mv hello.html lib
```

```
git add lib/hello.html
```

```
git rm hello.html
```

1.12.1 Коммит в новый каталог

Давайте сделаем коммит этого перемещения:

```
git commit -m "Moved hello.html to lib"
```

```

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ mkdir lib

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git mv hello.html lib/

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git status
On branch master
Changes to be committed:
(use "git restore --staged <file>..." to unstage)
    renamed:    hello.html -> lib/hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Moved hello.html to lib"
[master a23ab95] Moved hello.html to lib
 1 file changed, 0 insertions(+), 0 deletions(-)
 rename hello.html => lib/hello.html (100%)

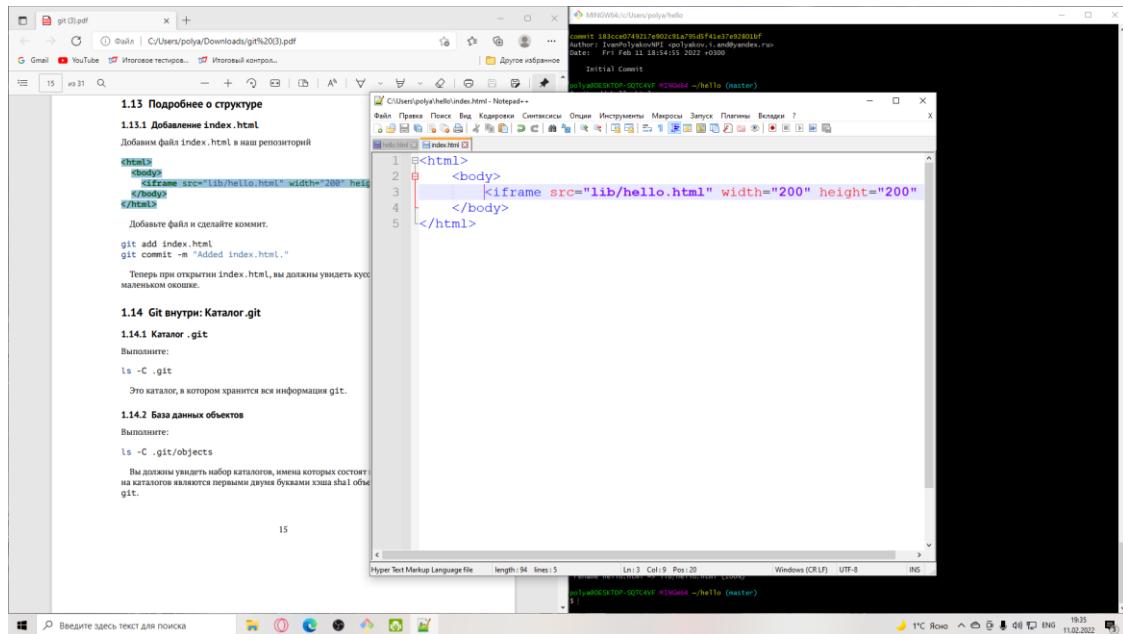
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ |

```

1.13 Подробнее о структуре

1.13.1 Добавление index.html

Добавим файл index.html в наш репозиторий



Добавьте файл и сделайте коммит.

`git add index.html`

`git commit -m "Added index.html."`

Теперь при открытии index.html, вы должны увидеть кусок страницы hello в маленьком окошке.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add index.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Added index.html."
[master ccb4fe2] Added index.html.
 1 file changed, 5 insertions(+)
 create mode 100644 index.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ |
```

1.14 Git внутри: Каталог .git

1.14.1 Каталог .git

Выполните:

```
ls -C .git
```

Это каталог, в котором хранится вся информация git.

1.14.2 База данных объектов

Выполните:

```
ls -C .git/objects
```

Вы должны увидеть набор каталогов, имена которых состоят из 2 символов. Имена каталогов являются первыми двумя буквами хэша sha1 объекта, хранящегося в git.

1.14.3 Углубляемся в базу данных объектов

Выполните:

```
ls -C .git/objects/
```

Смотрим в один из каталогов с именем из 2 букв. Вы увидите файлы с именами из 38 символов. Это файлы, содержащие объекты, хранящиеся в git. Они сжаты и закодированы, поэтому просмотр их содержимого нам мало чем поможет.

1.14.4 Config File

Выполните:

```
cat .git/config
```

Это файл конфигурации, создающийся для каждого конкретного проекта. Записи в этом файле будут перезаписывать записи в файле .gitconfig вашего главного каталога, по крайней мере в рамках этого проекта.

1.14.5 Ветки и теги

Выполните:

```
ls .git/refs
```

```
ls .git/refs/heads  
ls .git/refs/tags  
cat .git/refs/tags/v1
```

Вы должны узнавать файлы в подкаталоге тегов. Каждый файл соответствует тегу, ранее созданному с помощью команды git tag. Его содержание — это всего лишь хэш коммита, привязанный к тегу. Каталог heads практически аналогичен, но используется для веток, а не тегов. На данный момент у нас есть только одна ветка, так что все, что вы увидите в этом каталоге – это ветка master.

1.14.6 Файл HEAD

Выполните:

```
cat .git/HEAD
```

Файл HEAD содержит ссылку на текущую ветку, в данный момент это должна быть ветка master.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ ls -C .git  
COMMIT_EDITMSG HEAD ORIG_HEAD config description hooks/ index info/ logs/ objects/ packed-refs refs/  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ ls -C .git/objects/  
00/ 13/ 18/ 26/ 3c/ 41/ 50/ 66/ 73/ 86/ 9e/ a3/ bd/ cc/ d9/ e7/ pack/  
08/ 14/ 1d/ 39/ 3e/ 4b/ 52/ 6c/ 83/ 8a/ a2/ aa/ c2/ cd/ da/ info/  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ ls -C .git/objects/3c  
ea0f2fe95be8c8e331477edfa5f1c60b766ad3  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ cat .git/config  
[core]  
repositoryformatversion = 0  
filemode = false  
bare = false  
logallrefupdates = true  
symlinks = false  
ignorecase = true  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ ls .git/refs  
heads/ tags/  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ ls .git/refs/heads/  
master  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ ls .git/refs/tags/  
v1 v1-beta  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ cat .git/refs/tags/v1  
66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ cat .git/HEAD  
ref: refs/heads/master  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ |
```

1.15 Работа непосредственно с объектами git

##1.15.1 Поиск последнего коммита

Выполните:

```
git log -max-count=1
```

Эта команда должна показать последний коммит в репозиторий. SHA1 хэш в вашей системе, вероятно, отличается от моего, но вы увидите что-то наподобие этого.

1.15.2 Вывод последнего коммита с помощью SHA1 хэша

Выполните:

```
git cat-file -t
```

```
git cat-file -p
```

1.15.3 Поиск дерева

Мы можем вывести дерево каталогов, ссылка на который идет в коммите. Это должно быть описание файлов (верхнего уровня) в нашем проекте (для конкретного коммита). Используйте SHA1 хэш из строки «дерева», из списка выше. Выполните:

```
git cat-file -p
```

1.15.4 Вывод каталога lib

Выполните:

```
git cat-file -p
```

1.15.5 Вывод файла hello.html

Выполните:

```
git cat-file -p
```

1.15.6 Исследуйте самостоятельно

Исследуйте git репозиторий вручную самостоятельно. Смотрите, удастся ли вам найти оригинальный файл hello.html с самого первого коммита вручную по ссылкам SHA1 хэша в последнем коммите.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --max-count=1
commit ccb4fe2ac701009584c01df646201ef3a28a3766 (HEAD -> master)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:35:41 2022 +0300

    Added index.html.

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git cat-file -t ccb4fe2ac70100
commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git cat-file -p ccb4fe2ac70100
tree c2149b98f8989bf3e9b95d6dab4f5d0bf678c975
parent a23ab95e65748e5ea8dbff1620f6ba77f53fad79
author IvanPolyakovNPI <polyakov.i.and@yandex.ru> 1644597341 +0300
committer IvanPolyakovNPI <polyakov.i.and@yandex.ru> 1644597341 +0300

    Added index.html.

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git cat-file -p c2149b98f8989
100644 blob aa2d5ac2363aadb622c63e548fe1c339aa958ad2      index.html
040000 tree 262674c387897ea057a25e52cabf8b969f910a45      lib

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git cat-file -p 262674c387
100644 blob bdda2e6f5e03d90c6bc604d31625dc486273a088      hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git cat-file -p bdda2e6f5e03d9
<!-- Author: Polyakov I.A. (pochta1) -->
<html>
    <head>
    </head>
    <body>
        <h1>Hello, World!</h1>
    </body>
</html>
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git cat-file -t bdda2e6f5e03d9
blob

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$
```

1.16 Создание ветки

Пора сделать наш hello world более выразительным. Так как это может занять некоторое время, лучше переместить эти изменения в отдельную ветку, чтобы изолировать их от изменений в ветке master.

1.16.1 Создайте ветку

Давайте назовем нашу новую ветку «style». Выполните:

```
git checkout -b style
```

```
git status
```

git checkout -b является шорткатом для git branch

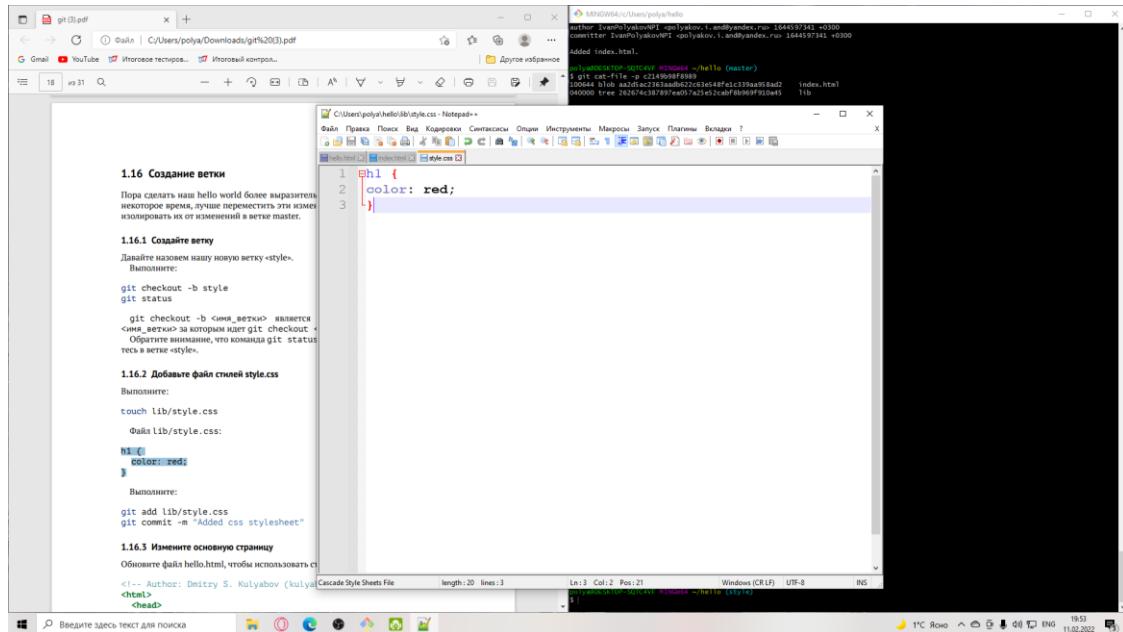
за которым идет git checkout . Обратите внимание, что команда git status сообщает о том, что вы находитесь в ветке «style».

1.16.2 Добавьте файл стилей style.css

Выполните:

```
touch lib/style.css
```

Файл lib/style.css:



```
1 h1 {
2   color: red;
3 }
```

Выполните:

```
git add lib/style.css
```

```
git commit -m "Added css stylesheet"
```

1.16.3 Измените основную страницу

Обновите файл hello.html, чтобы использовать стили style.css.

Below the code, there is a section of text in Russian: '1.16.4 Измените index.html' followed by instructions to update index.html to link to style.css and commit the changes. The bottom status bar shows the date as 11.02.2022."/>

```

<!DOCTYPE HTML>
<html>
  <head>
    <link type="text/css" rel="stylesheet"
      media="all" href="style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>

Былоните:
git add lib/hello.html
git commit -m "Hello uses style.css"

1.16.4 Измените index.html
Обновите файл index.html, чтобы он тоже использовал style.css
<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>

Былоните:
git add index.html
git commit -m "Updated index.html"

1.17 Навигация по веткам
Теперь в нашем проекте есть две ветки:
Вашоните:

```

Выполните:

`git add lib/hello.html`

`git commit -m "Hello uses style.css"`

1.16.4 Измените index.html

Обновите файл index.html, чтобы он тоже использовал style.css

Below the code, there is a section of text in Russian: '1.16.4 Измените index.html' followed by instructions to update index.html to link to style.css and commit the changes. The bottom status bar shows the date as 11.02.2022."/>

```

<!DOCTYPE HTML>
<html>
  <head>
    <link type="text/css" rel="stylesheet"
      media="all" href="lib/style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>

Былоните:
git add lib/hello.html
git commit -m "Hello uses style.css"

1.16.4 Измените index.html
Обновите файл index.html, чтобы он тоже использовал style.css
<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>

Былоните:
git add index.html
git commit -m "Updated index.html"

1.17 Навигация по веткам
Теперь в нашем проекте есть две ветки:
Вашоните:

```

Выполните:

`git add index.html`

`git commit -m "Updated index.html"`

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout -b style
Switched to a new branch 'style'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git status
On branch style
nothing to commit, working tree clean

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ touch lib/style.css

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git add lib/style.css

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git commit -m "Added css stylesheet"
[style 6dc1e5c] Added css stylesheet
 1 file changed, 3 insertions(+)
 create mode 100644 lib/style.css

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git add lib/hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git commit -m "Hello uses style.css"
On branch style
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   index.html

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

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

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git add index.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git commit -m "Updated index.html"
[style 43cdc11] Updated index.html
 1 file changed, 4 insertions(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$
```

1.17 Навигация по веткам

Теперь в вашем проекте есть две ветки: Выполните:

```
git log --all
```

1.17.1 Переключение на ветку master

Используйте команду git checkout для переключения между ветками:

```
git checkout master
```

```
cat lib/hello.html
```

Сейчас мы находимся на ветке master. Это заметно по тому, что файл hello.html не использует стили style.css.

1.17.2 Вернемся к ветке style

Выполните:

```
git checkout style
```

```
cat lib/hello.html
```

Содержимое lib/hello.html подтверждает, что мы вернулись на ветку style.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git log --pretty=oneline
79c622319a1bc2884a0042abd2c4cd8d356df877 (HEAD -> style) Hello uses style.css
43cdc11ca958ccc6361f551900dfffa5767746f88 Updated index.html
6dc1e5ca42beaae925f189904f6b8lef4a450629 Added css stylesheet
ccb4fe2ac701009584c01df646201ef3a28a3766 (master) Added index.html.
a23ab95e65748e5ea8dbff1620f6ba77f53fad79 Moved hello.html to lib
50157fdf2aacd527e47adb21688faafb813a46b0 Add an author/email comment
66a7b5d4a9bbeae81ef1508aa3f7b67f0cb0b648 (tag: v1) Added standard HTML page tags
e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta) Added standard HTML page tags
00288540c668b30590e79df71c159232995d1699 Initial Commit 2
183cce0749217e902c91a795d5f41e37e92801bf Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ cat lib/hello.html
<!-- Author: Polyakov I.A. (pochta1) -->
<html>
    <head>
    </head>
    <body>
        <h1>Hello, World!</h1>
    </body>
</html>
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout style
Switched to branch 'style'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ cat lib/hello.html
<!-- Author: Polyakov I.A. (pochta1) -->
<html>
    <head>
        <link type="text/css" rel="stylesheet"
            media="all" href="lib/style.css" />
    </head>
    <body>
        <h1>Hello, World!</h1>
    </body>
</html>
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ |
```

1.18 Изменения в ветке master

Пока вы меняли ветку style, кто-то решил обновить ветку master. Они добавили файл README.md.

1.18.1 Создайте файл README в ветке master

Выполните:

```
git checkout master
```

Создайте файл README.md

```
echo "This is the Hello World example from the git tutorial." > README.md
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ touch README.md

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ echo "This is the Hello World example from the git tutorial." > README.md

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$
```

1.19 Сделайте коммит изменений README.md в ветку master.

Выполните:

```
git add README.md
```

```
git commit -m "Added README"
```

1.19.1 Просмотр отличающихся веток

1.19.2 Просмотрите текущие ветки

Теперь у нас в репозитории есть две отличающиеся ветки. Используйте следующую лог-команду для просмотра веток и их отличий. Выполните:

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

Добавление опции `--graph` в `git log` вызывает построение дерева коммитов с помощью простых ASCII символов. Мы видим обе ветки (`style` и `master`), и то, что ветка `master` является текущей `HEAD`. Общим предшественником обеих веток является коммит «`Added index.html`». Опция `--all` гарантированно означает, что мы видим все ветки. По умолчанию показывается только текущая ветка.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --graph --all
* commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7 (HEAD -> master)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:08:34 2022 +0300
|
|     Added README
|
* commit 79c622319a1bc2884a0042abd2c4cd8d356df877 (style)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:04:37 2022 +0300
|
|     Hello uses style.css
|
* commit 43cdc11ca958ccc6361f551900dfffa5767746f88
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:56:48 2022 +0300
|
|     Updated index.html
|
* commit 6dc1e5ca42beaae925f189904f6b81ef4a450629
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:53:47 2022 +0300
|
|     Added css stylesheet
|
* commit ccb4fe2ac701009584c01df646201ef3a28a3766
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:35:41 2022 +0300
|
|     Added index.html.
|
* commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:33:55 2022 +0300
|
|     Moved hello.html to lib
|
* commit 50157fdf2aacd527e47adb21688faafb813a46b0
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:29:44 2022 +0300
|
|     Add an author/email comment
|
* commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:04:23 2022 +0300
|
|     Added standard HTML page tags
|
* commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:03:24 2022 +0300
|
|     Added standard HTML page tags
|
* commit 00288540c668b30590e79df71c159232995d1699
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 18:59:14 2022 +0300
|
|     Initial Commit 2
|
* commit 183cce0749217e902c91a795d5f41e37e92801bf
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 18:54:55 2022 +0300
|
|     Initial Commit
|
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
```

1.20 Слияние

1.20.1 Слияние веток

Слияние переносит изменения из двух веток в одну. Давайте вернемся к ветке style и сольем master с style. Выполните:

```
git checkout style
```

```
git merge master
```

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

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

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

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout style
Switched to branch 'style'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git merge master
Merge made by the 'ort' strategy.
 README.md | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git log --graph --all
* commit aa50e8f3a5472d3714e72a271ca10def00be59ab (HEAD -> style)
 |\ Merge: 79c6223 6dcae58
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 20:10:05 2022 +0300
 |
 |     Merge branch 'master' into style
 |
 * commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7 (master)
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 20:08:34 2022 +0300
 |
 |     Added README
 |
 * commit 79c622319a1bc2884a0042abd2c4cd8d356df877
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 20:04:37 2022 +0300
 |
 |     Hello uses style.css
 |
 * commit 43cdc11ca958ccc6361f551900dffaf5767746f88
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 19:56:48 2022 +0300
 |
 |     Updated index.html
 |
 * commit 6dc1e5ca42beeae925f189904f6b81ef4a450629
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 19:53:47 2022 +0300
 |
 |     Added css stylesheet
 |
 * commit ccb4fe2ac701009584c01df646201ef3a28a3766
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 19:35:41 2022 +0300
 |
 |     Added index.html.
 |
 * commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 19:33:55 2022 +0300
 |
 |     Moved hello.html to lib
 |
 * commit 50157fdf2aacd527e47adb21688faafb813a46b0
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 19:29:44 2022 +0300
 |
 |     Add an author/email comment
 |
 * commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
 | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
 | Date:   Fri Feb 11 19:04:23 2022 +0300
```

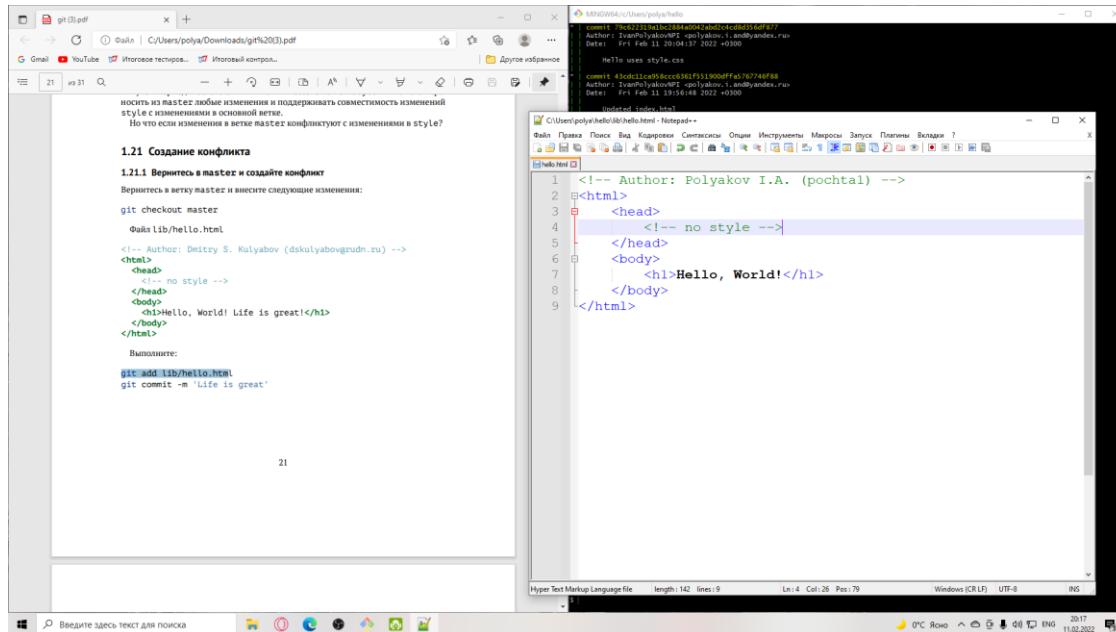
1.21 Создание конфликта

1.21.1 Вернитесь в master и создайте конфликт

Вернитесь в ветку master и внесите следующие изменения:

```
git checkout master
```

Файл lib/hello.html



Выполните:

```
git add lib/hello.html
```

```
git commit -m 'Life is great'
```

1.21.2 Просмотр веток

Выполните:

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

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. На следующем шаге мы решим этот конфликт.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m 'Life is great'
[master 36f3a1b] Life is great
 1 file changed, 1 insertion(+)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --graph --all
* commit 36f3a1b05381dc04d9971f45e18df126128c1409 (HEAD -> master)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:17:25 2022 +0300
|
|     Life is great
|
| * commit aa50e8f3a5472d3714e72a271ca10def00be59ab (style)
| | Merge: 79c6223 6dcae58
| | Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| | Date:   Fri Feb 11 20:10:05 2022 +0300
|
| |     Merge branch 'master' into style
|
* commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:08:34 2022 +0300
|
|     Added README
|
* commit 79c622319a1bc2884a0042abd2c4cd8d356df877
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:04:37 2022 +0300
|
|     Hello uses style.css
|
* commit 43cdc11ca958ccc6361f551900dfffa5767746f88
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:56:48 2022 +0300
|
|     Updated index.html
|
* commit 6dc1e5ca42beeae925f189904f6b81ef4a450629
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:53:47 2022 +0300
|
|     Added css stylesheet
|
* commit ccb4fe2ac701009584c01df646201ef3a28a3766
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:35:41 2022 +0300
|
|     Added index.html.
|
* commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:33:55 2022 +0300
```

1.22 Разрешение конфликтов

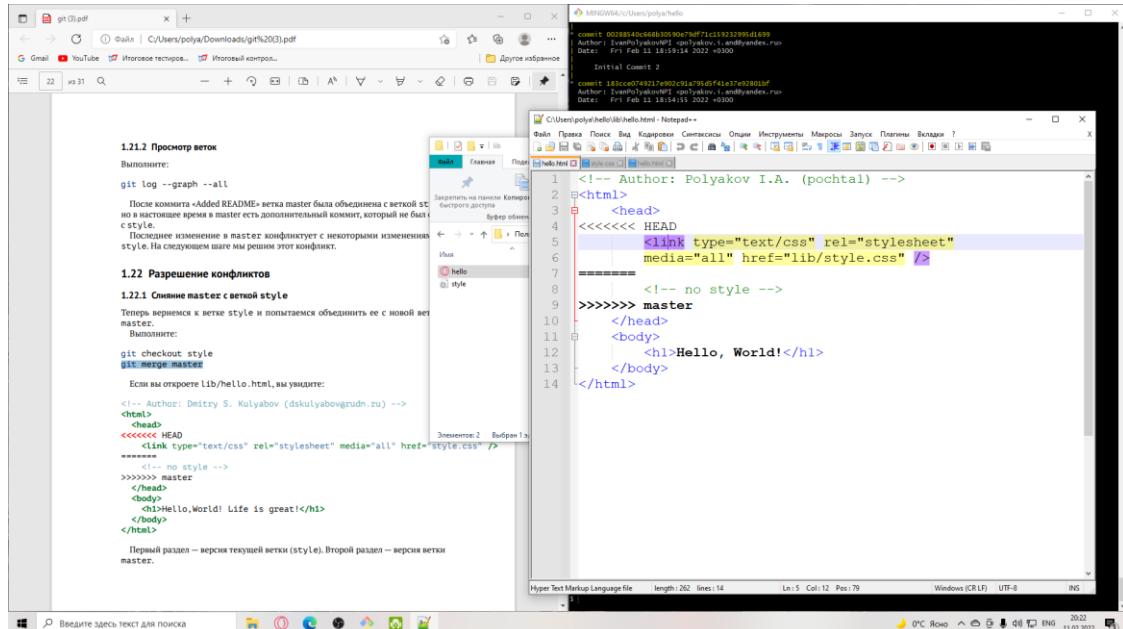
1.22.1 Слияние master с веткой style

Теперь вернемся к ветке style и попытаемся объединить ее с новой веткой master. Выполните:

```
git checkout style
```

git merge master

Если вы откроете lib/hello.html, вы увидите:



```
git log --graph -all
* commit 00288540c6d8b30590a7bdff71c159232895d1699
  Author: Ievgenii Polyakov <polyakov_i_and@yahoo.ru>
  Date: Fri Feb 11 18:15:12 2022 +0300
    Initial Commit ?
* commit 183cced0749217e0201a795df41e37e32801af
  Author: Ievgenii Polyakov <polyakov_i_and@yahoo.ru>
  Date: Fri Feb 11 18:15:15 2022 +0300
    Comit 1.1.1. Слияние мастер с веткой style
    Текущий коммит «Added README» ветка master была объединена с веткой style. На настоящий момент в мастер есть дополнительный коммит, который не был объединен.
    Последнее изменение в мастер конфликтует с некоторыми изменениями style. На следующем шаге мы решим этот конфликт.

1.2.2 Разрешение конфликтов
1.2.2.1 Слияние мастер с веткой style
Теперь вернемся к ветке style и попытаемся объединить ее с новой веткой master.
Выполните:
git checkout style
git merge master
Если вы откроете lib/hello.html, вы увидите:
<!-- Author: Dmitry S. Kulyabov (dskulayabov@rudn.ru) -->
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="style.css"/>
  </head>
  <body>
    <h1>Hello, World! Life is great!</h1>
  </body>
</html>
Первый раздел — первая текущей ветки (style). Второй раздел — версия ветки master.
```

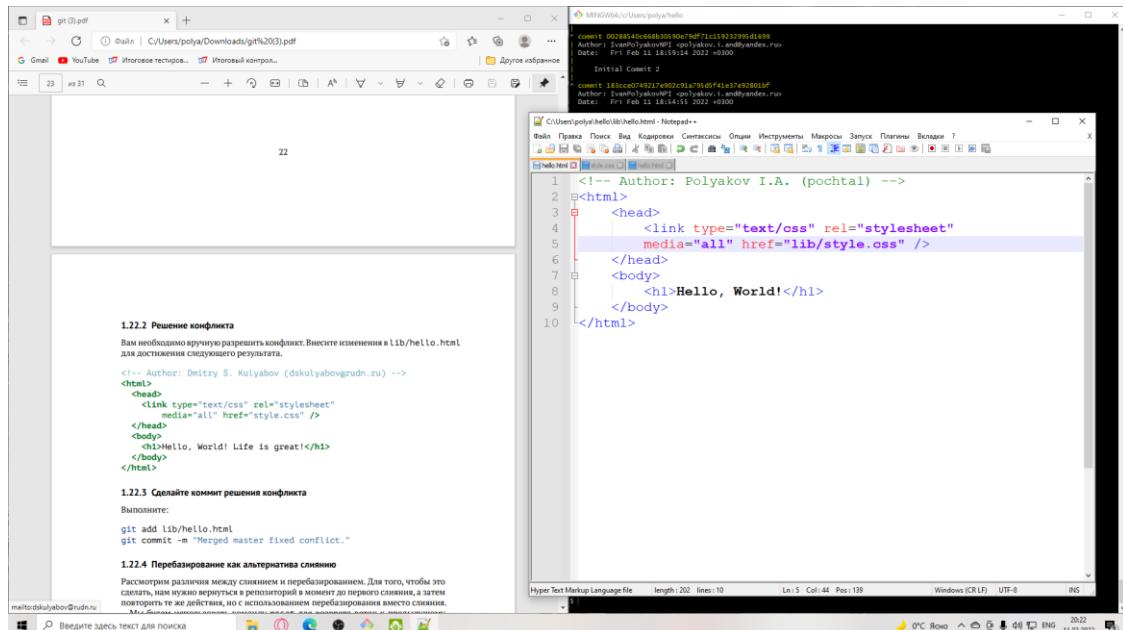
Содержимое файла lib/hello.html:

```
<!-- Author: Polyakov I.A. (pochtai) -->
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="lib/style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

Первый раздел — версия текущей ветки (style). Второй раздел — версия ветки master.

1.2.2.2 Решение конфликта

Вам необходимо вручную разрешить конфликт. Внесите изменения в lib/hello.html для достижения следующего результата.



```
git log --graph -all
* commit 00288540c6d8b30590a7bdff71c159232895d1699
  Author: Ievgenii Polyakov <polyakov_i_and@yahoo.ru>
  Date: Fri Feb 11 18:15:12 2022 +0300
    Initial Commit ?
* commit 183cced0749217e0201a795df41e37e32801af
  Author: Ievgenii Polyakov <polyakov_i_and@yahoo.ru>
  Date: Fri Feb 11 18:15:15 2022 +0300
    Comit 1.1.1. Слияние мастер с веткой style
    Текущий коммит «Added README» ветка master была объединена с веткой style. На настоящий момент в мастер есть дополнительный коммит, который не был объединен.
    Последнее изменение в мастер конфликтует с некоторыми изменениями style. На следующем шаге мы решим этот конфликт.

1.2.2.2 Решение конфликта
Выполните:
git checkout style
git merge master
Если необходимо вручную разрешить конфликт: Внесите изменения в lib/hello.html для достижения следующего результата.

<!-- Author: Dmitry S. Kulyabov (dskulayabov@rudn.ru) -->
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="style.css" />
  </head>
  <body>
    <h1>Hello, World! Life is great!</h1>
  </body>
</html>

1.2.2.3 Сделайте коммит решения конфликта
Выполните:
git add lib/hello.html
git commit -m "Merged master fixed conflict."
1.2.2.4 Перебазированием как альтернатива слиянию
Рассмотрим различия между слиянием и перебазированием. Для того, чтобы это сделать, нам нужно вернуться в реологию в момент до первого слияния, а затем повторить те же действия, но с использованием перебазирования вместо слияния.
...  
mailto:dskulayabov@rudn.ru
```

Содержимое файла lib/hello.html:

```
<!-- Author: Polyakov I.A. (pochtai) -->
<html>
  <head>
    <link type="text/css" rel="stylesheet" media="all" href="lib/style.css" />
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

1.2.2.3 Сделайте коммит решения конфликта

Выполните:

```
git add lib/hello.html
```

```
git commit -m "Merged master fixed conflict."
```

1.22.4 Перебазирование как альтернатива слиянию

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.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git merge master
Auto-merging lib/hello.html
CONFLICT (content): Merge conflict in lib/hello.html
Automatic merge failed; fix conflicts and then commit the result.

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style|MERGING)
$ git add lib/hello.html

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style|MERGING)
$ git commit -m "Merged master fixed conflict."
[style 8324248] Merged master fixed conflict.
```

1.23 Сброс ветки style

1.23.1 Сброс ветки style

Вернемся на ветку `style` к точке перед тем, как мы слили ее с веткой `master`. Мы можем сбросить ветку к любому коммиту. По сути, это изменение указателя ветки на любую точку дерева коммитов. В этом случае мы хотим вернуться в ветку `style` в точку перед слиянием с `master`. Нам необходимо найти последний коммит перед слиянием.

Выполните:

```
git checkout style
```

```
git log --graph
```

Мы видим, что коммит «Updated index.html» был последним на ветке `style` перед слиянием. Давайте сбросим ветку `style` к этому коммиту. Выполните:

```
git reset --hard
```

1.23.2 Проверьте ветку.

Поиските лог ветки `style`. У нас в истории больше нет коммитов слияний. Выполните:

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

```
* commit 478f296354dadad5a307a3b0da3c8bcf08eafe54 (master)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:21:30 2022 +0300
|
|     Life is great2
|
* commit 36f3a1b05381dc04d9971f45e18df126128c1409
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:17:25 2022 +0300
|
|     Life is great
|
* commit 6dcae5825ce55e34ddbc533835eaac5e5f53cf7a
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:08:34 2022 +0300
|
|     Added README
|
* commit 43cdc11ca958ccc6361f551900dfffa5767746f88 (HEAD -> style)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:56:48 2022 +0300
|
|     Updated index.html
|
* commit 6dc1e5ca42beaae925f189904f6b81ef4a450629
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:53:47 2022 +0300
|
|     Added css stylesheet
|
* commit ccb4fe2ac701009584c01df646201ef3a28a3766
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:35:41 2022 +0300
|
|     Added index.html.
|
* commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:33:55 2022 +0300
|
|     Moved hello.html to lib
|
* commit 50157fdf2aacd527e47adb21688faafb813a46b0
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:29:44 2022 +0300
|
|     Add an author/email comment
|
* commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:04:23 2022 +0300
|
|     Added standard HTML page tags
|
* commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:03:24 2022 +0300
|
|     Added standard HTML page tags
|
* commit 00288540c668b30590e79df71c159232995d1699
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 18:59:14 2022 +0300
|
|     Initial Commit 2
|
* commit 183cce0749217e902c91a795d5f41e37e92801bf
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 18:54:55 2022 +0300
```

```
Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git reset --hard 6dcae5825ce55e
HEAD is now at 6dcae58 Added README

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log --graph --all
* commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7 (HEAD -> master)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 20:08:34 2022 +0300
|
|     Added README
|
* commit 43cdc11ca958ccc6361f551900dffaf5767746f88 (style)
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:56:48 2022 +0300
|
|     Updated index.html
|
* commit 6dc1e5ca42beaae925f189904f6b81ef4a450629
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:53:47 2022 +0300
|
|     Added css stylesheet
|
* commit ccb4fe2ac701009584c01df646201ef3a28a3766
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:35:41 2022 +0300
|
|     Added index.html.
|
* commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
| Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
| Date:   Fri Feb 11 19:33:55 2022 +0300
|
|     Moved hello.html to lib
```

1.24 Сброс ветки master

1.24.1 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. Выполните:

```
git checkout master
```

```
git log --graph
```

Коммит «Added README» идет непосредственно перед коммитом конфликтующего интерактивного режима. Мы сбросим ветку master к коммиту «Added README». Выполните:

```
git reset --hard
```

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

Просмотрите лог. Он должен выглядеть, как будто репозиторий был перемотан назад во времени к точке до какого-либо слияния.

1.25 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. Выполните:

```
git checkout style
```

```
git rebase master
```

```
git log --graph
```

1.25.1 Слияние VS перебазирование

Конечный результат перебазирования очень похож на результат слияния. Ветка `style` в настоящее время содержит все свои изменения, а также все изменения ветки `master`. Однако, дерево коммитов значительно отличается. Дерево коммитов ветки `style` было переписано таким образом, что ветка `master` является частью истории коммитов. Это делает цепь коммитов линейной и гораздо более читабельной. Не используйте перебазирование:

- если ветка является публичной и расшаренной, поскольку переписывание общих веток будет мешать работе других членов команды;
- когда важна точная история коммитов ветки, так как команда `rebase` переписывает историю коммитов;

Учитывая приведенные выше рекомендации, рекомендуется использовать `git rebase` для кратковременных, локальных веток, а слияние для веток в публичном репозитории.

1.26 Слияние в ветку `master`

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

1.26.1 Слияние `style` в `master`

Выполните:

```
git checkout master
```

```
git merge style
```

Поскольку последний коммит ветки `master` прямо предшествует последнему коммиту ветки `style`, `git` может выполнить ускоренное слияние-перемотку. При быстрой перемотке вперед `git` просто передвигает указатель вперед, таким образом указывая на тот же коммит, что и ветка `style`. При быстрой перемотке конфликтов быть не может.

1.26.2 Просмотрите логи

Выполните:

```
git log
```

Теперь ветки style и master идентичны.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (style)
$ git checkout master
Switched to branch 'master'

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git merge style
Updating 6dcae58..d736f02
Fast-forward
 index.html    | 4 +---
 lib/style.css | 3 +++
 2 files changed, 7 insertions(+)
 create mode 100644 lib/style.css

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git log
commit d736f0252cf3e8a4e006cce816255211e8d9bb7 (HEAD -> master, style)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:56:48 2022 +0300

    Updated index.html

commit a1fd4230e7cfc8b42057749f0291218b3a4b2d7d
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:53:47 2022 +0300

    Added css stylesheet

commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 20:08:34 2022 +0300

    Added README

commit ccb4fe2ac701009584c01df646201ef3a28a3766
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:35:41 2022 +0300

    Added index.html.

commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:33:55 2022 +0300

    Moved hello.html to lib

commit 50157fdf2aacd527e47adb21688faafb813a46b0
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:29:44 2022 +0300

    Add an author/email comment

commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags
```

1.27 Клонирование репозиториев

1.27.1 Перейдите в рабочий каталог

Перейдите в рабочий каталог и сделайте клон вашего репозитория hello. Выполните:

```
cd ..
```

```
pwd
```

```
ls
```

Сейчас мы находимся в рабочем каталоге. В этот момент вы должны находиться в «рабочем» каталоге. Здесь должен быть единственный репозиторий под названием «hello».

1.27.2 Создайте клон репозитория hello

Создадим клон репозитория. Выполните:

```
git clone hello cloned_hello
```

```
ls
```

В вашем рабочем каталоге теперь должно быть два репозитория: оригинальный репозиторий «hello» и клонированный репозиторий «cloned_hello»

```

polya@DESKTOP-SQTC4VF MINGW64 ~
$ ls
001-025.zip
'3D Objects'
Anaconda3/
AndroidStudioProjects/
AppData/
'Application Data'@
ClionProjects/
'Cisco Packet Tracer 7.2.2'/
Contacts/
Cookies@
Documents/
Downloads/
'Dprok addon 2.rar'
Favorites/
IdeaProjects/
Links/
'Local Settings'@
Minion.exe*
Minion.ico
Music/
NTUSER.DAT
NTUSER.DAT{b9d891a9-f49a-11eb-b690-7085c28d08dd}.TM.blf
NTUSER.DAT{b9d891a9-f49a-11eb-b690-7085c28d08dd}.TMContainer00000000000000000001.regtrans-ms
NTUSER.DAT{b9d891a9-f49a-11eb-b690-7085c28d08dd}.TMContainer00000000000000000002.regtrans-ms
NetHood@
OneDrive/
PrintHood@
Recent@
ST_lab1/
'Saved Games'/
Searches/
SendTo@
Vagrantfile
Videos/
'VirtualBox VMs'/
ansel/
app/
hello/
lab1/
msvcp100.dll*
msvcp120.dll*
msvcr100.dll*
msvcr120.dll*
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
packager.dll*
runtime/
signal/
source/
spectre/
unins000.dat
unins000.exe*
vagrant/
'Mои документы'@
'Снимки экрана'/
Шаблоны@
'главное меню'@

polya@DESKTOP-SQTC4VF MINGW64 ~
$ git clone hello cloned_hello
Cloning into 'cloned_hello'...
done.

polya@DESKTOP-SQTC4VF MINGW64 ~
$ ls
001-025.zip
'3D Objects'
Anaconda3/
AndroidStudioProjects/
AppData/
'Application Data'@
ClionProjects/
'Cisco Packet Tracer 7.2.2'/
Contacts/
Cookies@
Documents/
Downloads/
'Dprok addon 2.rar'
Favorites/
IdeaProjects/
Links/
'Local Settings'@
Minion.exe*
Minion.ico
Music/
NTUSER.DAT
NTUSER.DAT{b9d891a9-f49a-11eb-b690-7085c28d08dd}.TM.blf
NTUSER.DAT{b9d891a9-f49a-11eb-b690-7085c28d08dd}.TMContainer00000000000000000001.regtrans-ms
NTUSER.DAT{b9d891a9-f49a-11eb-b690-7085c28d08dd}.TMContainer00000000000000000002.regtrans-ms
NetHood@
OneDrive/
PrintHood@
Recent@
ST_lab1/
'Saved Games'/
Searches/
SendTo@
Vagrantfile
Videos/
'VirtualBox VMs'/
ansel/
app/
cloned_hello/
hello/
lab1/
msvcp100.dll*
msvcp120.dll*
msvcr100.dll*
msvcr120.dll*
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
packager.dll*
runtime/
signal/
source/
spectre/
unins000.dat
unins000.exe*
vagrant/
'Mои документы'@
'Снимки экрана'/
Шаблоны@
'главное меню'@

```

1.28 Просмотр клонированного репозитория

1.28.1 Давайте взглянем на клонированный репозиторий.

Выполните:

`cd cloned_hello`

`ls`

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

1.28.2 Просмотрите историю репозитория

Выполните:

```
git log -all
```

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

1.28.3 Удаленные ветки

Вы увидите ветку master (HEAD) в списке истории. Вы также увидите ветки со странными именами (origin/master, origin/style и origin/HEAD).

```
polya@DESKTOP-SQTC4VF MINGW64 ~
$ cd cloned_hello/
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ ls
README.md index.html lib/
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git log --all
commit d736f0252cf3e8a4e006cce816255211e8d9bb7 (HEAD -> master, origin/style, origin/master, origin/HEAD)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:56:48 2022 +0300

    Updated index.html

commit a1fd4230e7cfc8b42057749f0291218b3a4b2d7d
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:53:47 2022 +0300

    Added css stylesheet

commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 20:08:34 2022 +0300

    Added README

commit ccb4fe2ac701009584c01df646201ef3a28a3766
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:35:41 2022 +0300

    Added index.html.

commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:33:55 2022 +0300

    Moved hello.html to lib

commit 50157fdf2aacd527e47adb21688faafb813a46b0
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:29:44 2022 +0300

    Add an author/email comment

commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

    Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git remote
```

1.29 Что такое origin?

Выполните:

```
git remote
```

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: Выполните:

```
git remote show origin
```

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).

```
        Added css stylesheet

commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 20:08:34 2022 +0300

        Added README

commit ccb4fe2ac701009584c01df646201ef3a28a3766
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:35:41 2022 +0300

        Added index.html.

commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:33:55 2022 +0300

        Moved hello.html to lib

commit 50157fdf2aacd527e47adb21688faafb813a46b0
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:29:44 2022 +0300

        Add an author/email comment

commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

        Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

        Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

        Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

        Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git remote
origin

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git remote show origin
* remote origin
  Fetch URL: C:/Users/polya/hello
  Push  URL: C:/Users/polya/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)

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
```

1.30 Удаленные ветки

Давайте посмотрим на ветки, доступные в нашем клонированном репозитории. Выполните:

```
git branch
```

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

1.30.1 Список удаленных веток

Для того, чтобы увидеть все ветки, попробуйте следующую команду:

```
git branch -a
```

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

```
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git branch
* master

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/master
  remotes/origin/style

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$
```

1.31 Изменение оригинального репозитория

Внесите некоторые изменения в оригиналный репозиторий, чтобы затем попытаться извлечь и слить изменения из удаленной ветки в текущую

1.31.1 Внесите изменения в оригиналный репозиторий hello

Выполните:

```
cd ..\hello
```

Примечание: Сейчас мы находимся в репозитории hello Внесите следующие изменения в файл README.md: Файл README.md

This is the Hello World example from the git tutorial.

Теперь добавьте это изменение и сделайте коммит Выполните:

```
git add README
```

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

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

1.31.2 Извлечение изменений

Научиться извлекать изменения из удаленного репозитория. Выполните:

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

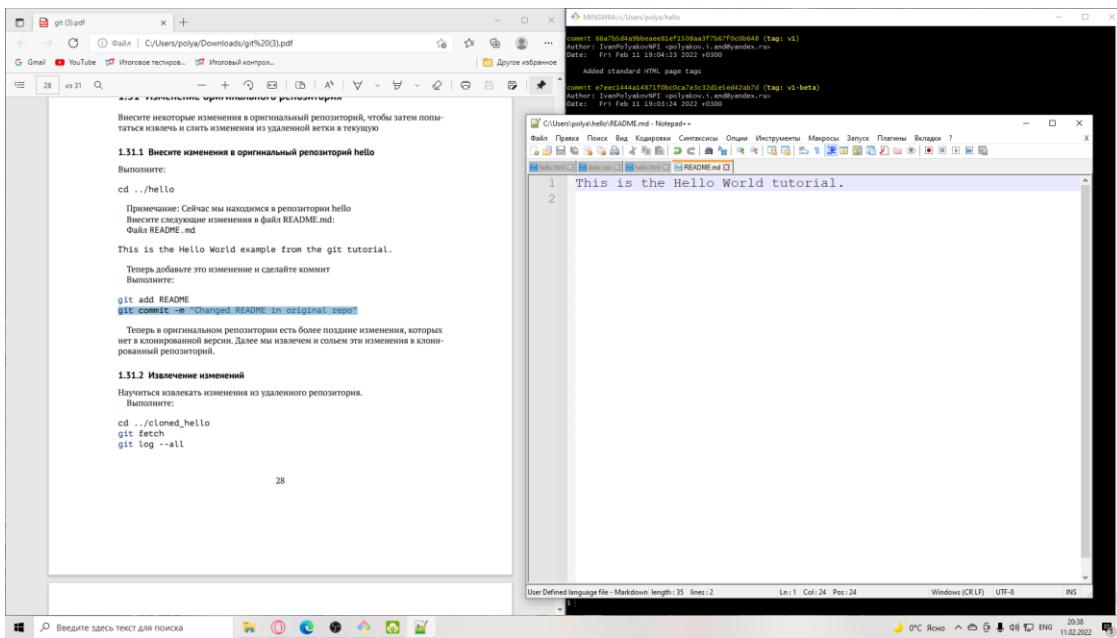
Сейчас мы находимся в репозитории cloned_hello. На данный момент в репозитории есть все коммиты из оригинального репозитория, но они не интегрированы в локальные ветки клонированного репозитория. В истории выше найдите коммит «Changed README in original repo». Обратите внимание, что коммит включает в себя коммиты «origin/master» и «origin/HEAD». Теперь давайте посмотрим на коммит «Updated index.html». Вы увидите, что локальная ветка master указывает на этот коммит, а не на новый коммит, который мы только что извлекли. Выводом является то, что команда git fetch будет извлекать новые коммиты из удаленного репозитория, но не будет сливать их с вашими наработками в локальных ветках.

1.31.3 Проверьте README.md

Мы можем продемонстрировать, что клонированный файл README.md не изменился. Выполните:

```
cat README
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)  
$ cd ../hello  
  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ git add README  
fatal: pathspec 'README' did not match any files  
  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ git add README.md  
  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ git commit -m "Changed README in original repo"  
[master 32bbce8] Changed README in original repo  
1 file changed, 1 insertion(+), 1 deletion(-)  
  
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)  
$ |
```



```
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git log --all
commit 32bbce8ddac2d39b7a2f287ec2256455439be2a8 (origin/master, origin/HEAD)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 20:38:18 2022 +0300

    Changed README in original repo

commit d736f0252cf3e8a4e006cce816255211e8d9bb7 (HEAD -> master, origin/style)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:56:48 2022 +0300

    Updated index.html

commit a1fd4230e7cfc8b42057749f0291218b3a4b2d7d
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:53:47 2022 +0300

    Added css stylesheet

commit 6dcae5825ce55e34ddbc533835eaac5e5f53cfa7
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 20:08:34 2022 +0300

    Added README

commit ccb4fe2ac701009584c01df646201ef3a28a3766
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:35:41 2022 +0300

    Added index.html.

commit a23ab95e65748e5ea8dbff1620f6ba77f53fad79
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:33:55 2022 +0300

    Moved hello.html to lib

commit 50157fdf2aacd527e47adb21688faafb813a46b0
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:29:44 2022 +0300

    Add an author/email comment

commit 66a7b5d4a9bbeaee81ef1508aa3f7b67f0c0b648 (tag: v1)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:04:23 2022 +0300

    Added standard HTML page tags

commit e7eec1444a14871f0bc0ca7e3c32d1e5ed42ab7d (tag: v1-beta)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:03:24 2022 +0300

    Added standard HTML page tags

commit 00288540c668b30590e79df71c159232995d1699
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:59:14 2022 +0300

    Initial Commit 2

commit 183cce0749217e902c91a795d5f41e37e92801bf
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 18:54:55 2022 +0300

    Initial Commit

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ cat README.md
This is the Hello World example from the git tutorial
```

1.32 Слияние извлеченных изменений

1.32.1 Слейте извлеченные изменения в локальную ветку master

Выполните:

```
git merge origin/master
```

1.32.2 Еще раз проверьте файл README.md

Сейчас мы должны увидеть изменения. Выполните:

```
cat README.md
```

Хотя команда git fetch не сливает изменения, мы можем вручную слить изменения из удаленного репозитория. Теперь давайте рассмотрим объединение fetch и merge в одну команду. Выполнение:

```
git pull
```

эквивалентно двум следующим шагам:

```
git fetch
```

```
git merge origin/master
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git merge origin/master
Updating d736f02..32bbce8
Fast-forward
 README.md | 2 ++
 1 file changed, 1 insertion(+), 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ cat README.md
This is the Hello World tutorial.

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git pull
Already up to date.

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$
```

1.33 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.

1.33.1 Добавьте локальную ветку, которая отслеживает удаленную ветку

Выполните:

```
git branch -track style origin/style
```

```
git branch -a
```

```
git log --max-count=2
```

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

```
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git branch --track style origin/style
branch 'style' set up to track 'origin/style'.

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git branch -a
* master
  style
    remotes/origin/HEAD -> origin/master
    remotes/origin/master
    remotes/origin/style

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git log --max-count=2
commit 32bbce8ddac2d39b7a2f287ec2256455439be2a8 (HEAD -> master, origin/master, origin/HEAD)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 20:38:18 2022 +0300

    Changed README in original repo

commit d736f0252cf3e8a4e006cce816255211e8d9bb7 (origin/style, style)
Author: IvanPolyakovNPI <polyakov.i.and@yandex.ru>
Date:   Fri Feb 11 19:56:48 2022 +0300

    Updated index.html

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
```

1.34 Чистые репозитории

Чистые репозитории (без рабочих каталогов) обычно используются для расширения. Обычный git-репозиторий подразумевает, что вы будете использовать его как рабочую директорию, поэтому вместе с файлами проекта в актуальной версии, git хранит все служебные, «чисто-репозиториевые» файлы в поддиректории .git. В удаленных репозиториях нет смысла хранить рабочие файлы на диске (как это делается в рабочих копиях), а все что им действительно нужно — это дельты изменений и другие бинарные данные репозитория. Вот это и есть «чистый репозиторий».

1.35 Создайте чистый репозиторий

```
cd ..
```

```
git clone --bare hello hello.git
```

```
ls hello.git
```

Сейчас мы находимся в рабочем каталоге. Как правило, репозитории, оканчивающиеся на .git являются чистыми репозиториями. Мы видим, что в репозитории hello.git нет рабочего каталога. По сути, это есть не что иное, как каталог .git нечистого репозитория.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ cd ..

polya@DESKTOP-SQTC4VF MINGW64 ~
$ git clone --bare hello hello.git
Cloning into bare repository 'hello.git'...
done.

polya@DESKTOP-SQTC4VF MINGW64 ~
$ ls hello.git
HEAD config description hooks/ info/ objects/ packed-refs refs/
polya@DESKTOP-SQTC4VF MINGW64 ~
$ |
```

1.36 Добавление удаленного репозитория

Давайте добавим репозиторий hello.git к нашему оригинальному репозиторию.

```
cd hello
```

```
git remote add shared ../hello.git
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~
$ cd hello

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git remote add shared ../hello.git

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ |
```

1.37 Отправка 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. Отредактируйте файл README.md и сделайте коммит Файл README.md:

```
This is the Hello World example from the git tutorial.
```

```
(Changed in the original and pushed to shared)
```

Выполните:

```
git checkout master
```

```
git add README
```

```
git commit -m "Added shared comment to readme"
```

Теперь отправьте изменения в общий репозиторий. Выполните:

```
git push shared master
```

Общим называется репозиторий, получающий отправленные нами изменения.

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git checkout master
Already on 'master'
M      README.md

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git add README.md

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git commit -m "Added shared comment to readme"
[master 25cd243] Added shared comment to readme
 1 file changed, 2 insertions(+), 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ git push shared master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 412 bytes | 412.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To ../hello.git
  32bbce8..25cd243  master -> master

polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ |
```

1.38 Извлечение общих изменений

Научиться извлекать изменения из общего репозитория. Быстро переключитесь в клонированный репозиторий и извлеките изменения, только что отправленные в общий репозиторий.

Выполните:

```
cd .../cloned_hello
```

Сейчас мы находимся в репозитории cloned_hello. Выполните:

```
git remote add shared ../hello.git
```

```
git branch -track shared master
```

```
git pull shared master
```

```
cat README.md
```

```
polya@DESKTOP-SQTC4VF MINGW64 ~/hello (master)
$ cd ../cloned_hello

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git remote add shared ../hello.git

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git branch --track shared master
branch 'shared' set up to track 'master'.

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ git pull shared master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 392 bytes | 3.00 KiB/s, done.
From ../hello
 * branch           master      -> FETCH_HEAD
 * [new branch]     master      -> shared/master
Updating 32bbce8..25cd243
Fast-forward
 README.md | 3 +++
 1 file changed, 2 insertions(+), 1 deletion(-)

polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ cat README.md
This is the Hello World example from the git tutorial.
(Changed in the original and pushed to shared)
polya@DESKTOP-SQTC4VF MINGW64 ~/cloned_hello (master)
$ |
```

Выходы

Научился работать с системой контроля версий Git.