# Работа с Git

## Подготовка

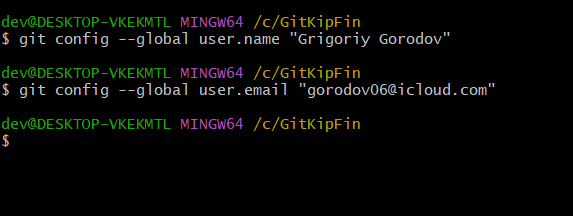


Рисунок 1 - Установка имени и электронной почты

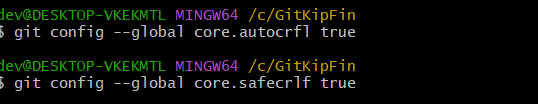


Рисунок 2 - Параметры окончаний строк

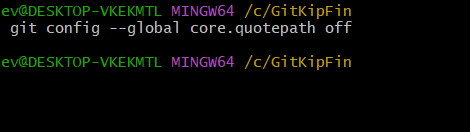


Рисунок 3 - Установка отображения unicode

## Создание проекта

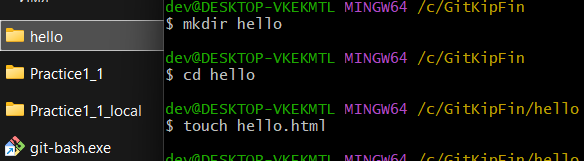


Рисунок 4 - Создание каталога и файла

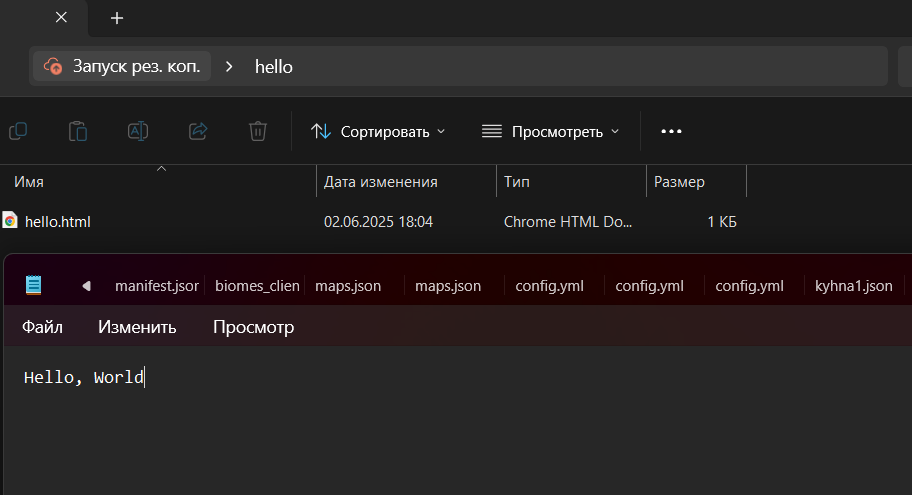


Рисунок 5 - Содержание файла

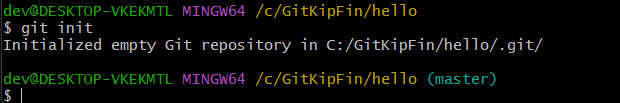


Рисунок 6 - Создание репозитория

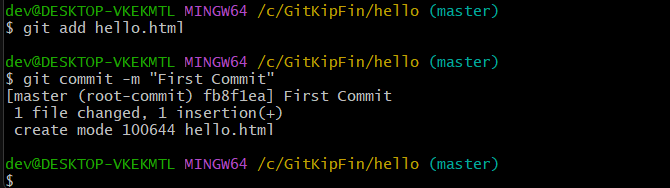


Рисунок 7 - Добавление в репозиторий

## Проверка состояния

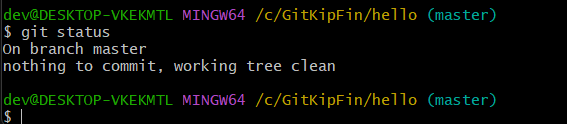


Рисунок 8 - Проверка состояния репозитория

## Внесение изменений

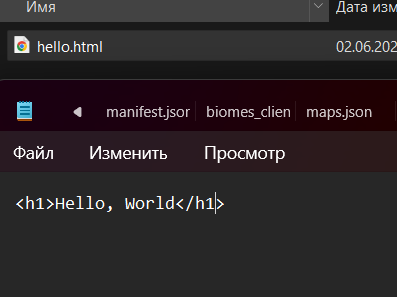


Рисунок 9 - Внесение изменений в файл

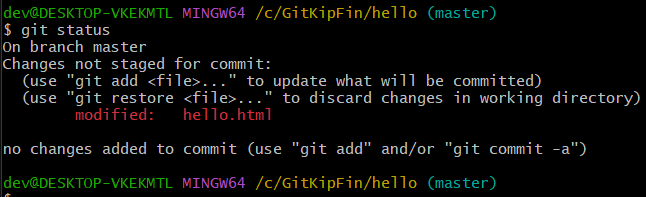


Рисунок 10 - Сообщение о незафиксированных изменениях

## Индексация изменений

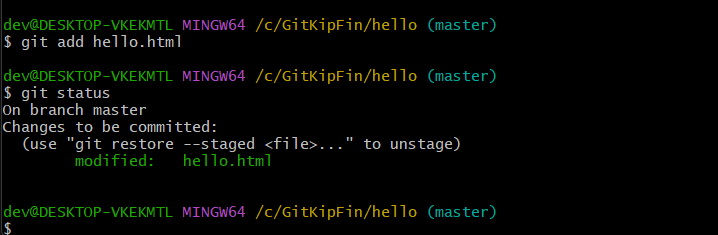


Рисунок 11 - Команды для индексации изменений

## Индексация и коммит

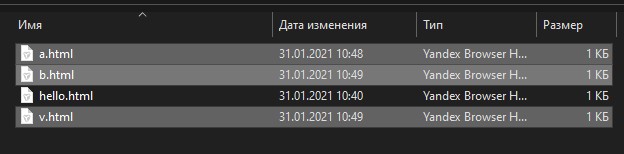


Рисунок 12 - Создано 3 файла

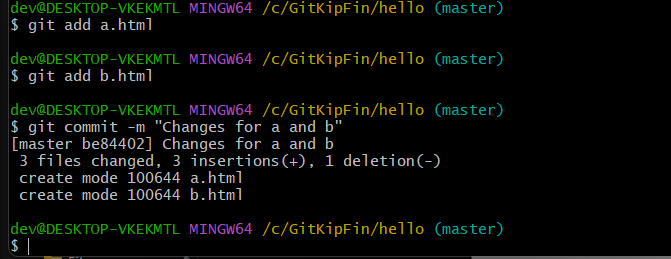


Рисунок 13 - Индексация и коммит для 2 файлов

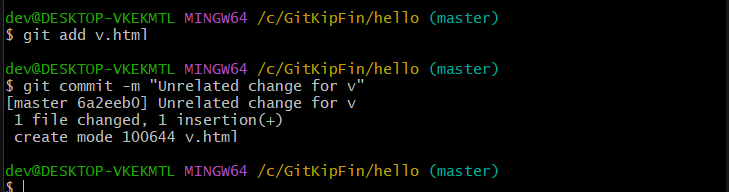


Рисунок 14 - Индексация и коммит для третьего файла

## Коммит изменений



Рисунок 15 - Коммит изменений

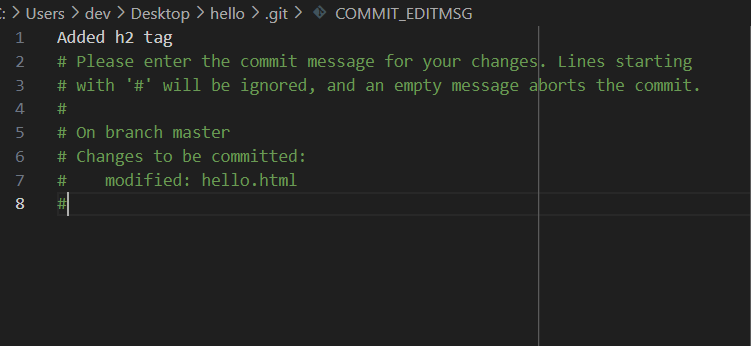


Рисунок 16 - Ввод комментария



Рисунок 17 - Коммит-сообщения

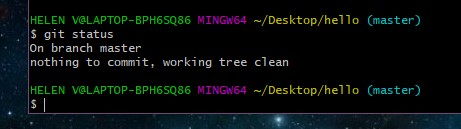


Рисунок 18 - Проверка состояния

## Изменения, а не файлы

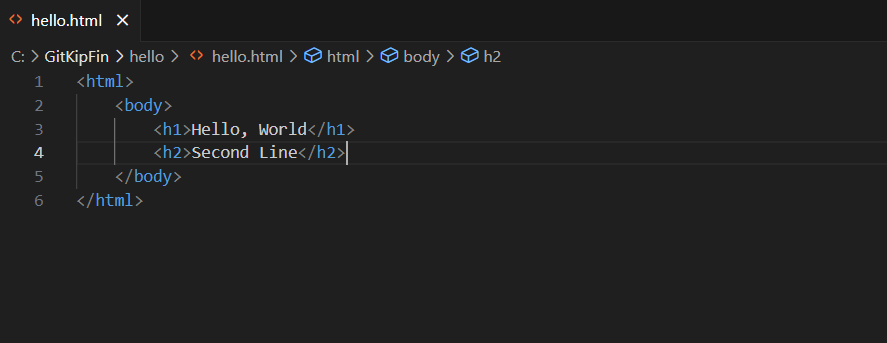


Рисунок 19 - Добавление тегов html и body

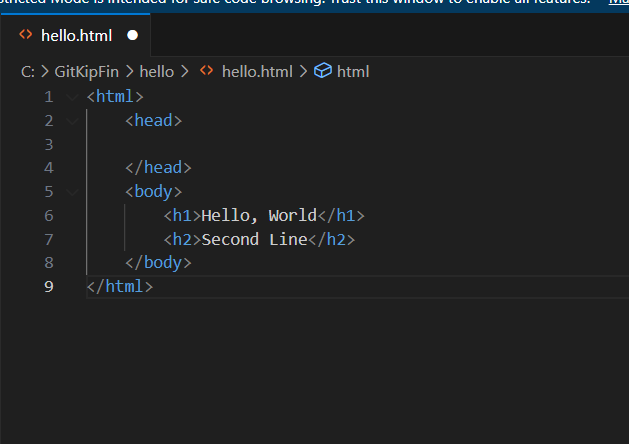


Рисунок 20 - Добавление тега head

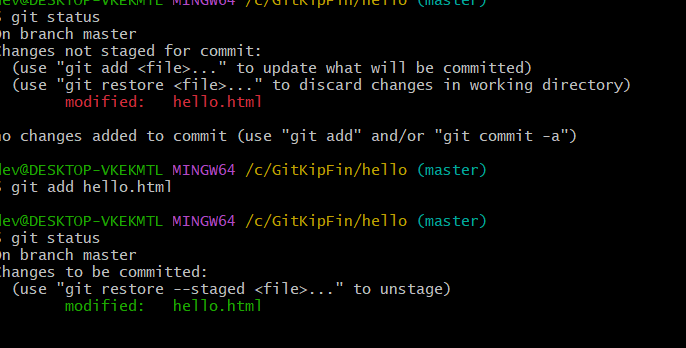


Рисунок 21 - Проверка состояния

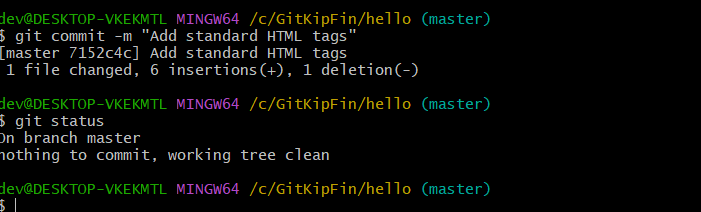


Рисунок 22 - Коммит 1 изменения и проверка состояния

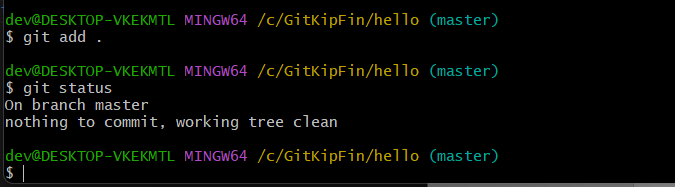


Рисунок 23 - Индексация 2 изменения и проверка состояния

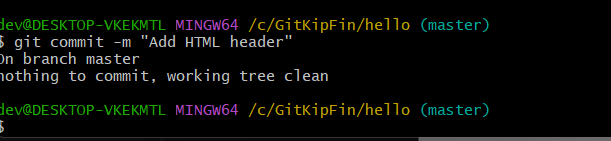


Рисунок 24 - Коммит 2 изменения

# Работа с Git

## История



Рисунок 25 - Просмотр истории изменений

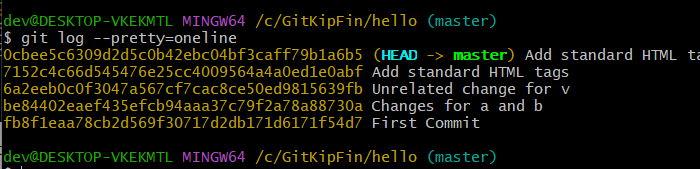


Рисунок 6 - Однострочный формат вывода

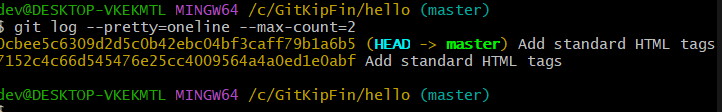


Рисунок 27 - Вывод последних 2 изменений

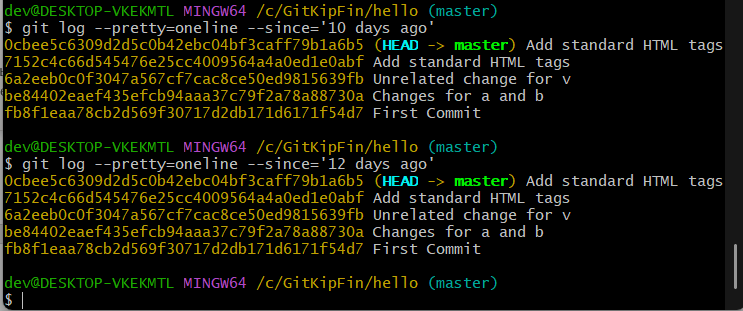


Рисунок 28 - Вывод изменений начиная с определенного времени

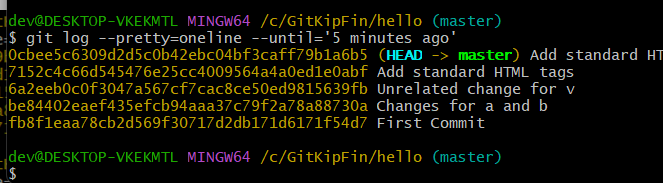


Рисунок 29 - Вывод изменений до определенного времени

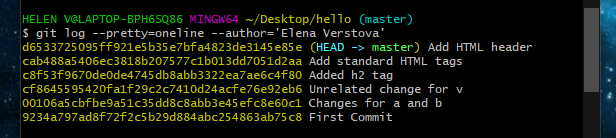


Рисунок 30 - Вывод изменений, внесенных определенным автором

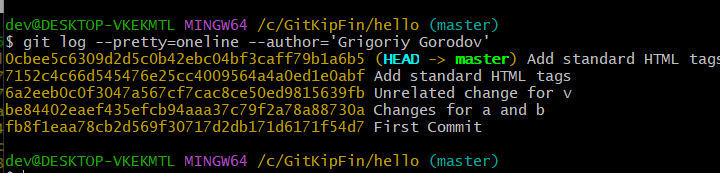


Рисунок 31 - Вывод всех изменений

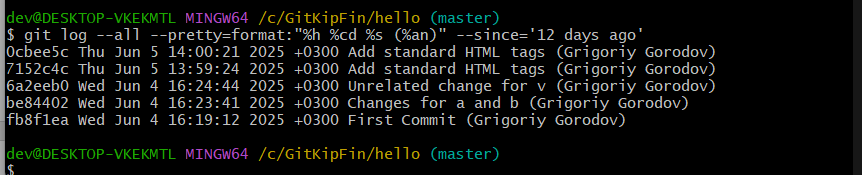


Рисунок 32 - Использование нескольких параметров

## Алиасы



Рисунок 33 - Настройка алиасов для некоторых команд

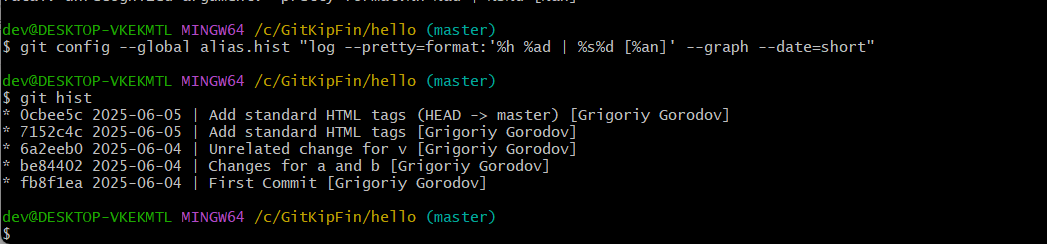


Рисунок 34 - Выполнение алиаса hist

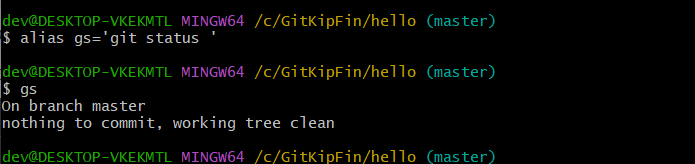


Рисунок 35 - Установка и выполнение алиаса gs

## Получение старых версий

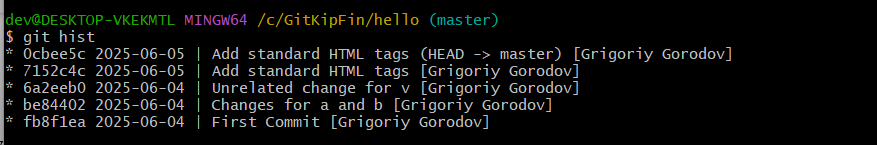


Рисунок 36 - Просмотр хэшей предыдущих версий

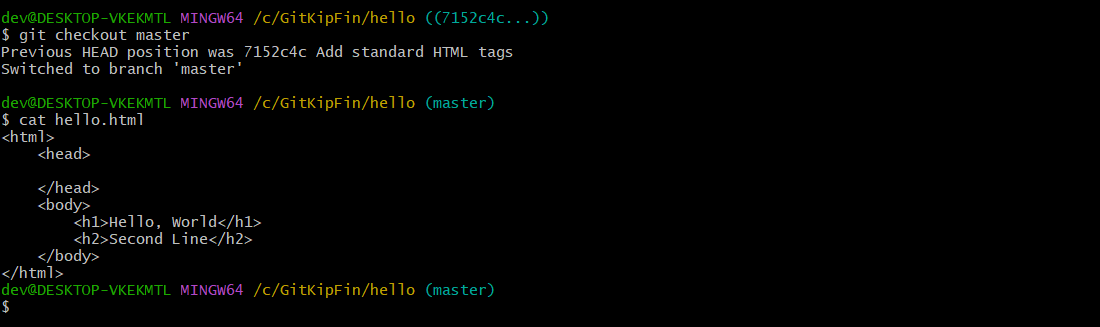


Рисунок 37 - Возвращение к нужной версии и просмотр содержимого файла

## Возвращение к последней версии в ветке master

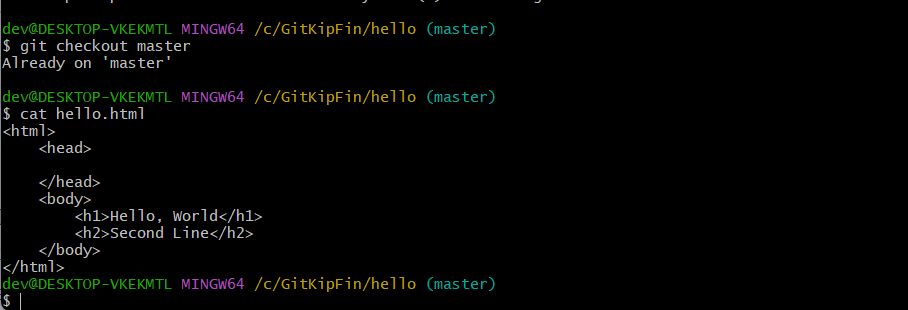


Рисунок 38 - Возвращение к последней версии в ветке master

## Создание тегов версий

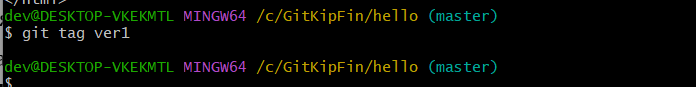


Рисунок 39 - Задание тега

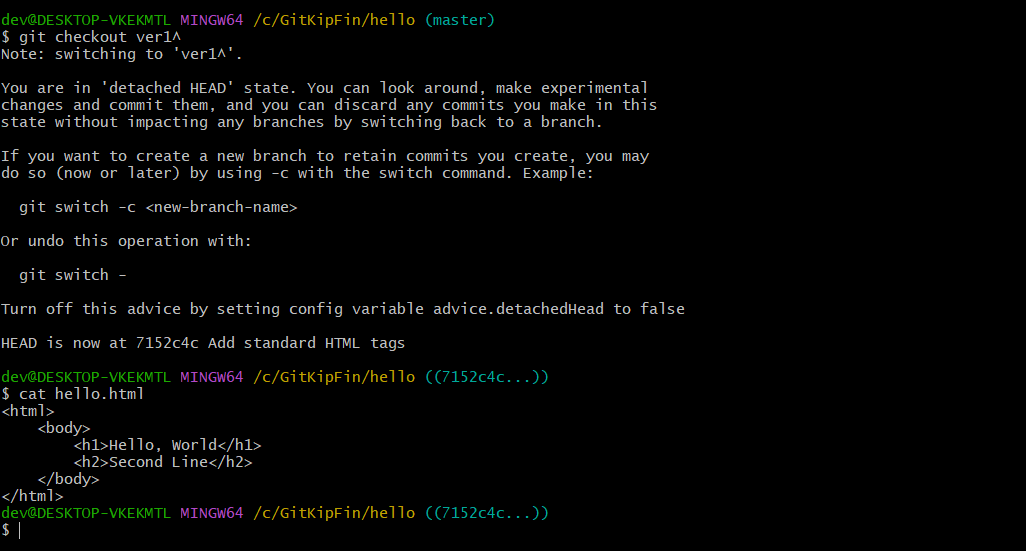


Рисунок 40 – Переход к предыдущей версии с помощью тега

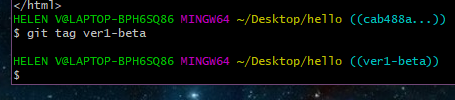


Рисунок 41 - Задание тега предыдущей версии

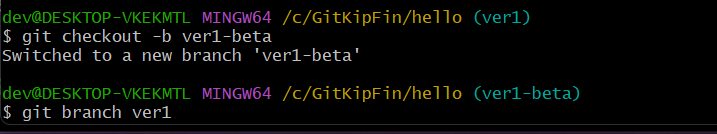


Рисунок 42 - Переключение между версиями с помощью тегов

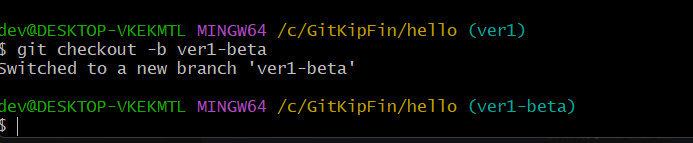


Рисунок 43 - Просмотр тегов

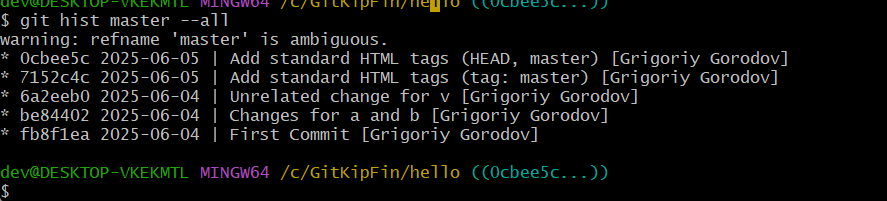


Рисунок 44 - Просмотр тегов в логе

## Отмена локальных изменений (до индексации)

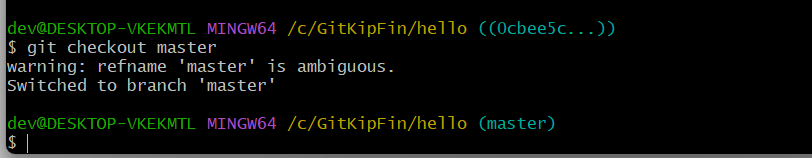


Рисунок 45 - Переключение на последний коммит

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

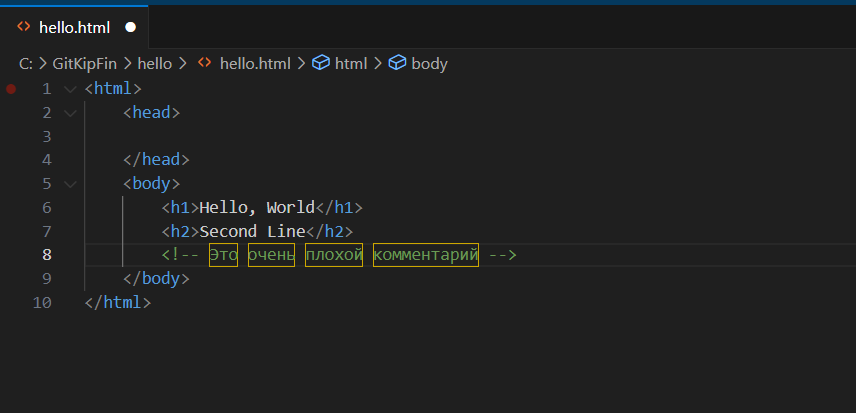


Рисунок 46 - Внесение изменения в файл

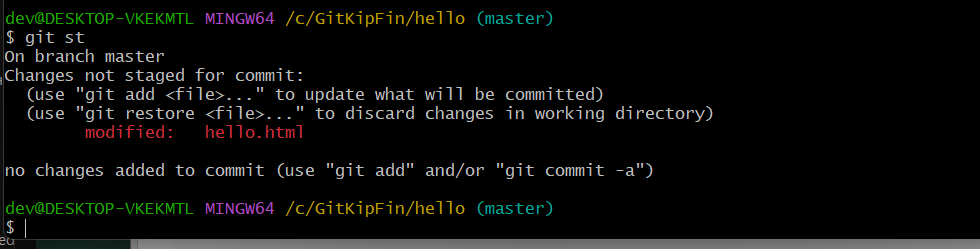


Рисунок 47 - Не проиндексированное изменение

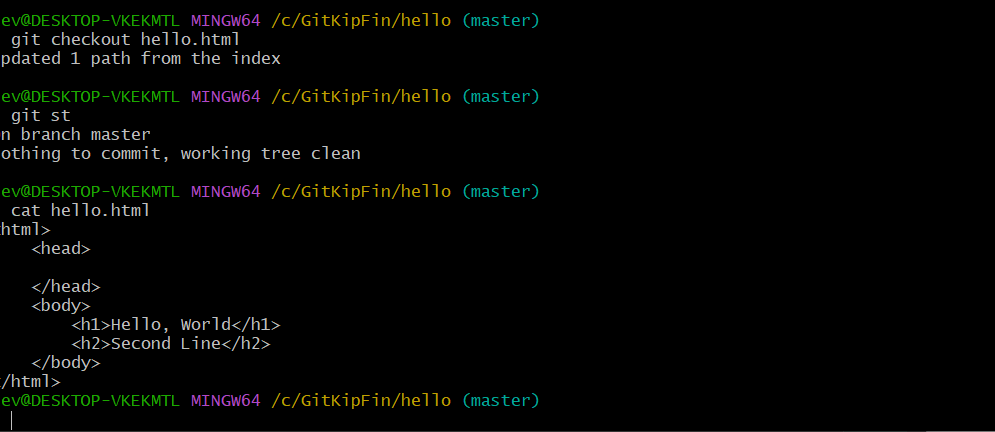


Рисунок 8 - Возвращение к версии

## Отмена проиндексированных изменений (перед коммитом)

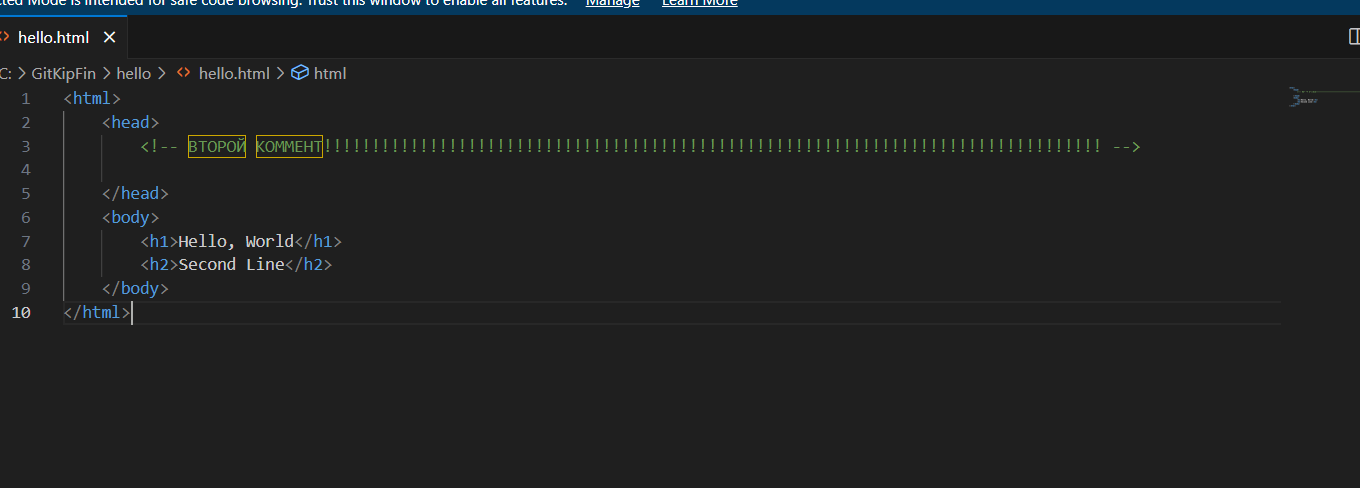


Рисунок 49 - Внесение ненужного изменения

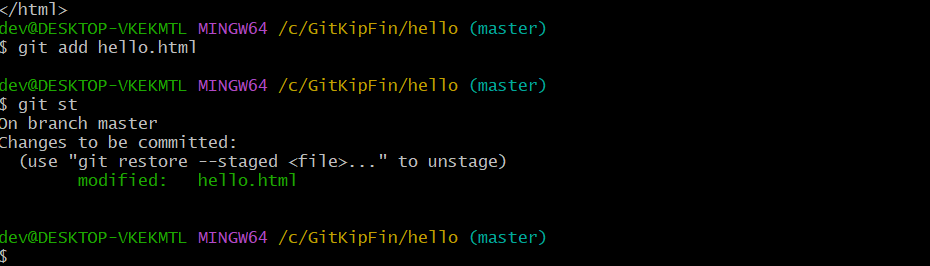


Рисунок 50 - Индексация изменения

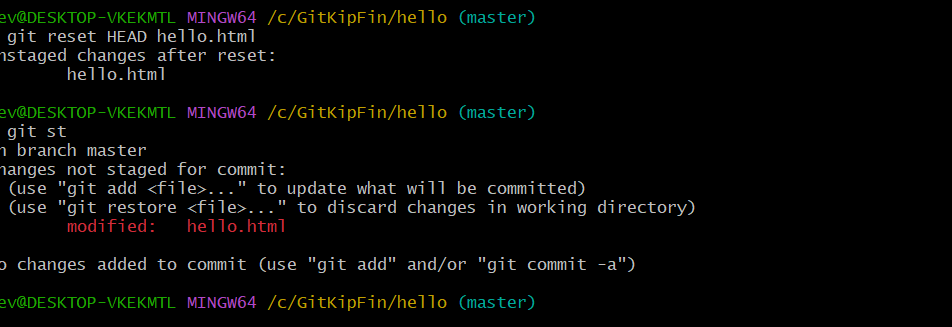


Рисунок 51 - Очистка буферной зоны

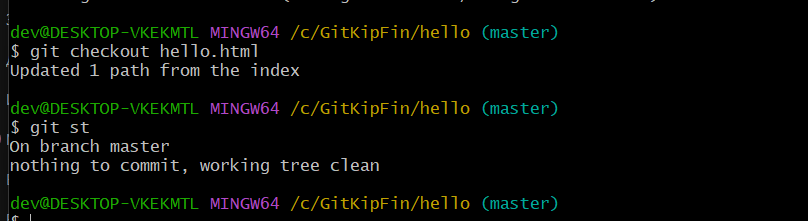


Рисунок 52 - Удаление ненужных изменений

## Отмена коммитов

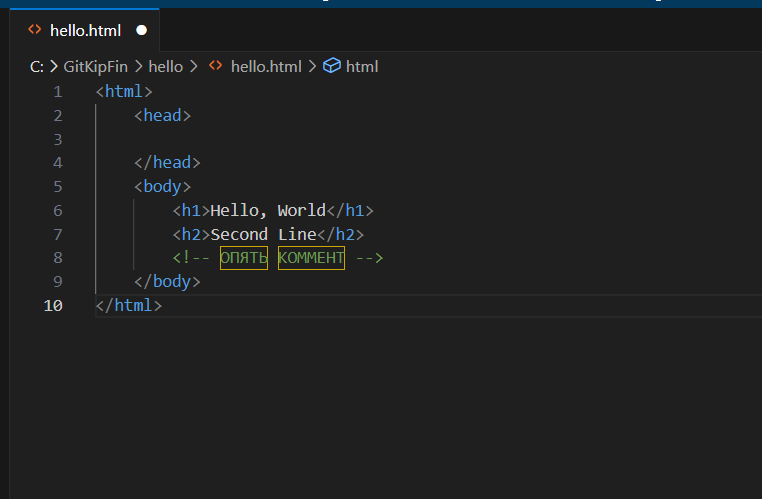


Рисунок 53 - Внесение изменения в файл

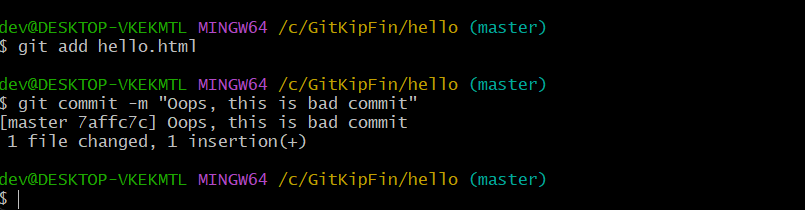


Рисунок 54 - Индексация и коммит

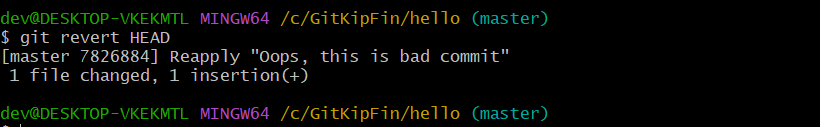


Рисунок 55 - Выполнение команды git revert

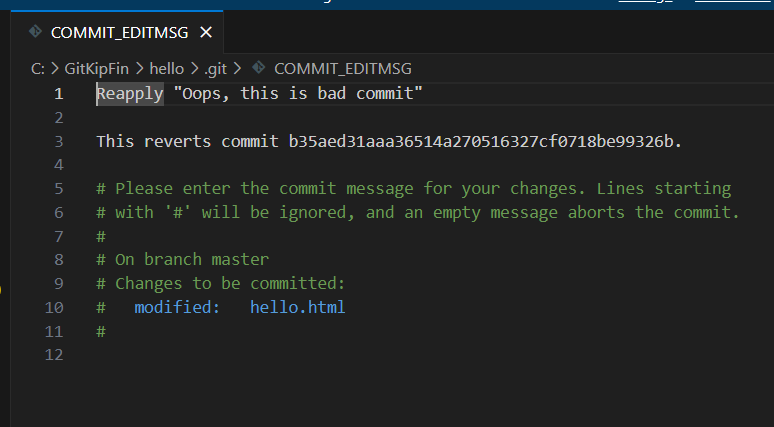


Рисунок 56 - Коммит сообщение в редакторе

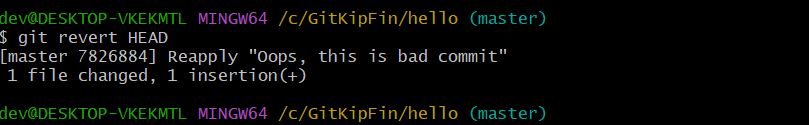


Рисунок 57 - Редактор закрыт

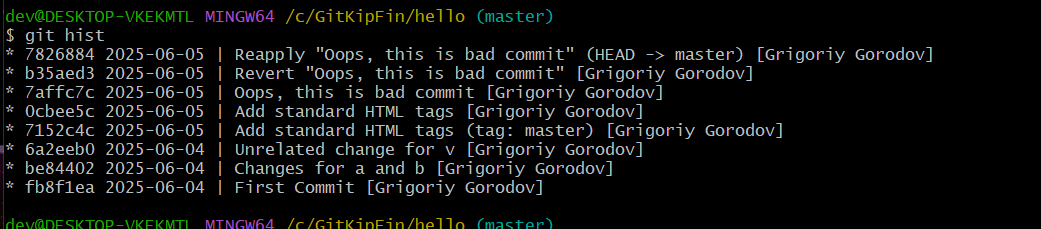


Рисунок 58 - Все коммиты при просмотре лога



Рисунок 59 - Отметка тегом коммита

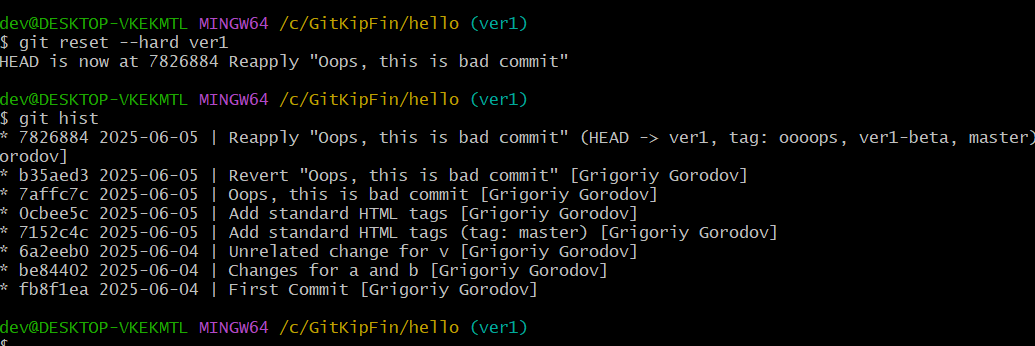


Рисунок 60 - Сброс коммита

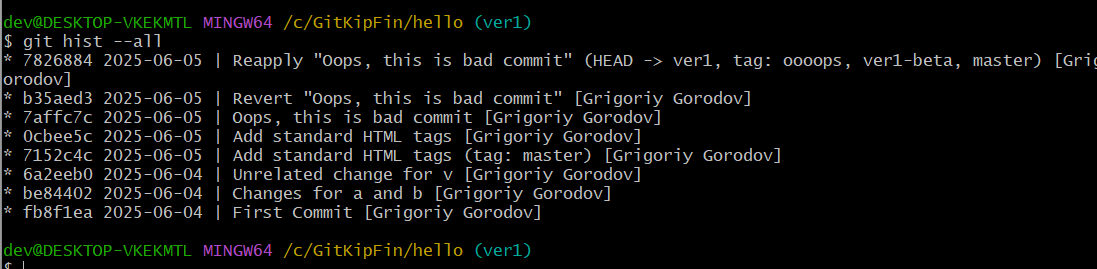


Рисунок 61 - Сброшенные коммиты находятся по-прежнему в репозитории

## Удаление тега

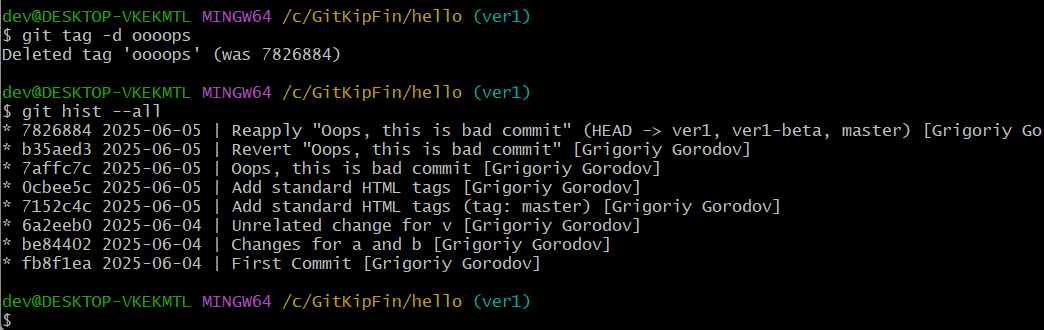


Рисунок 62 - Удаление тега

# Работа с Git

## Внесение изменений в коммиты

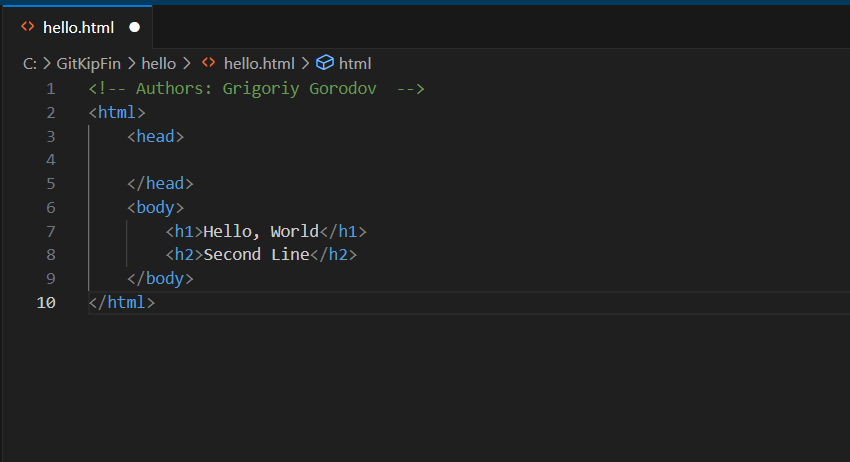


Рисунок 63 - Добавление комментария в файл

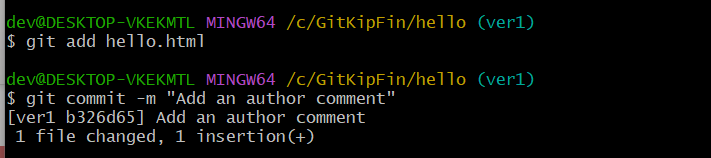


Рисунок 64 - Индексация и коммит

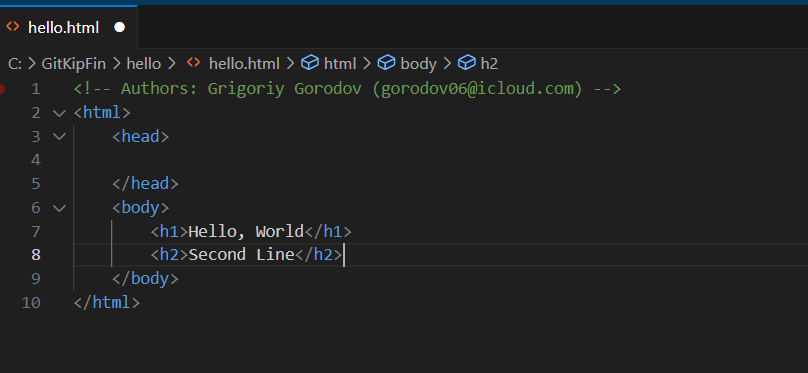


Рисунок 65 - Добавление электронной почты

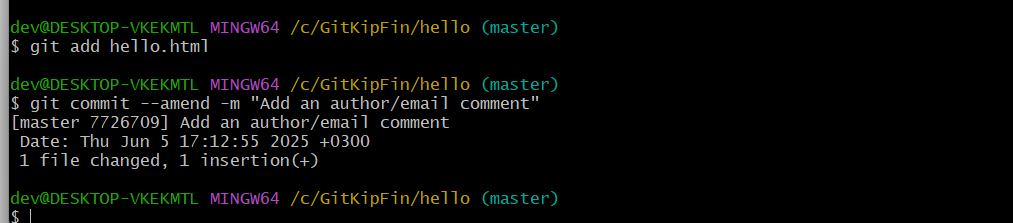


Рисунок 66 - Индексация и изменение коммита



Рисунок 67 - Последний коммит изменен

## Перемещение файлов

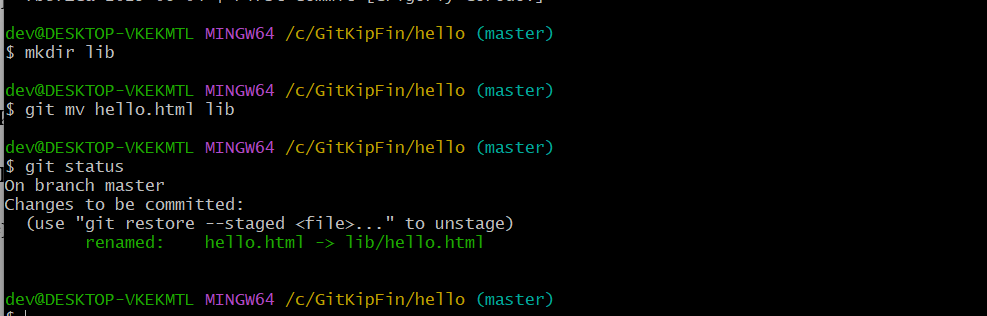


Рисунок 8 - Перемещение файла

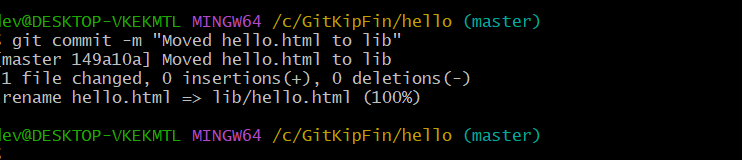


Рисунок 69 – Коммит перемещения

## Подробнее о структуре

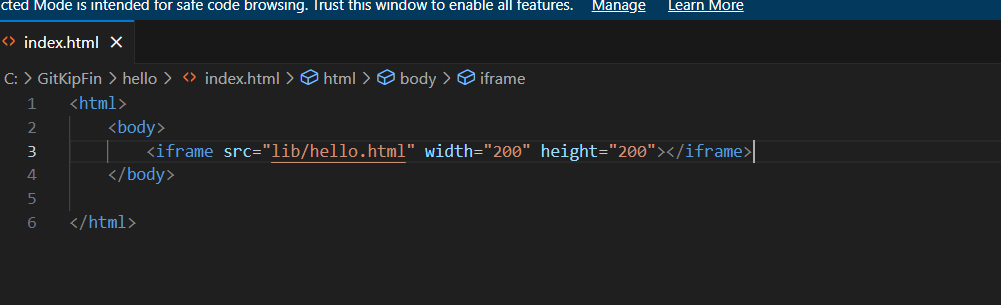


Рисунок 70 - Содержимое файла index.html

Далее нужно проиндексировать и закоммитить файл (рисунок 9).

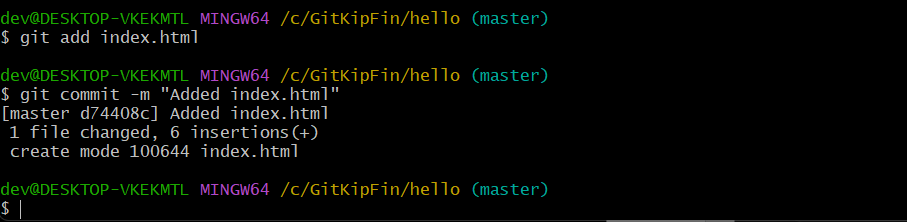


Рисунок 71 - Индексация и коммит

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



Рисунок 72 - Файл index.html, открытый в браузере

## Каталог .git

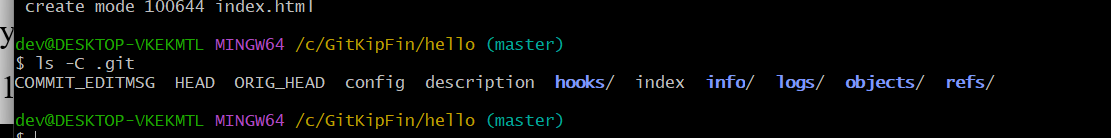


Рисунок 73 - Содержание каталога .git

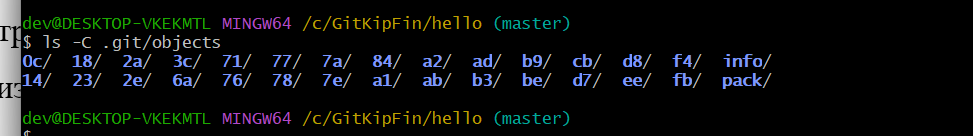


Рисунок 74 - Содержание каталога objects

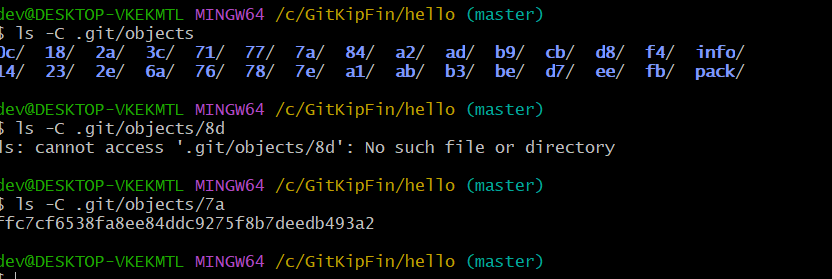


Рисунок 75 - Просмотр каталога 8d

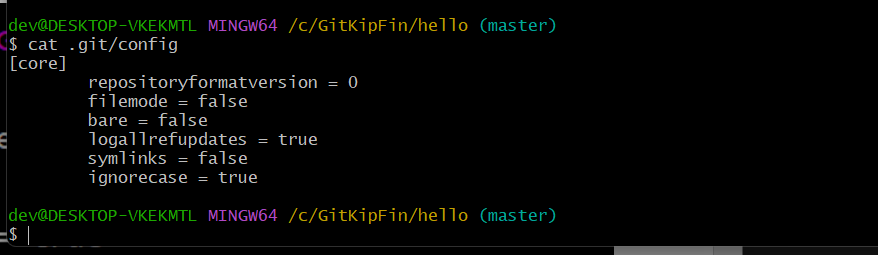


Рисунок 76 - Просмотр файла конфигурации

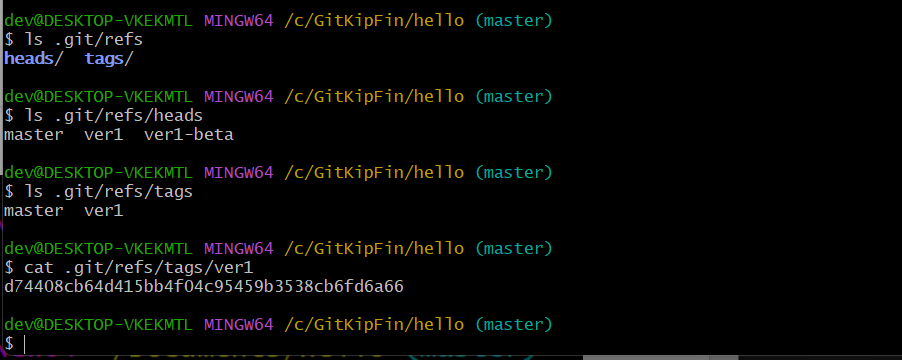


Рисунок 77 - Просмотр файлов и веток

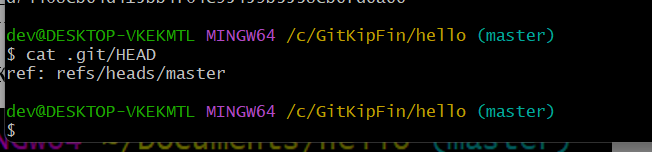


Рисунок 78 - Содержимое файла HEAD

## Работа с объектами git

Для начала необходимо просмотреть последний коммит (рисунок 17).

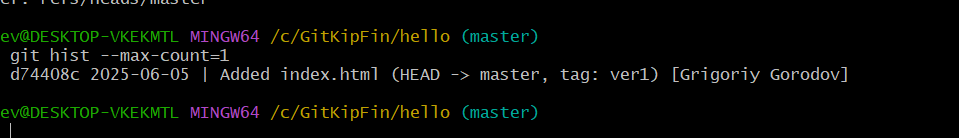


Рисунок 79 - Последний коммит



Рисунок 0 - Просмотр объекта коммита

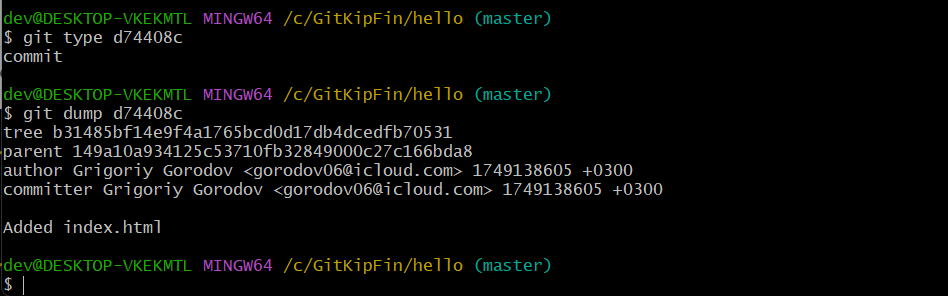


Рисунок 81 - Использование алиасов

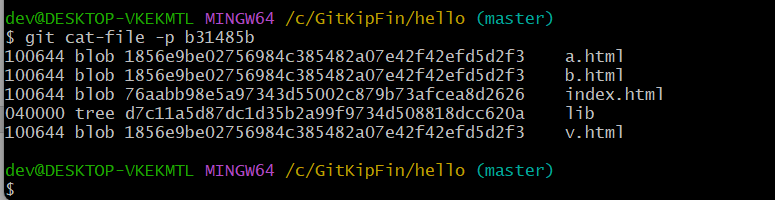


Рисунок 82 - Просмотр дерева каталогов

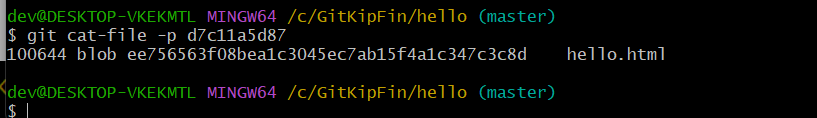


Рисунок 83 - Просмотр каталога lib



Рисунок 84 - Вывод содержимого файла hello.html

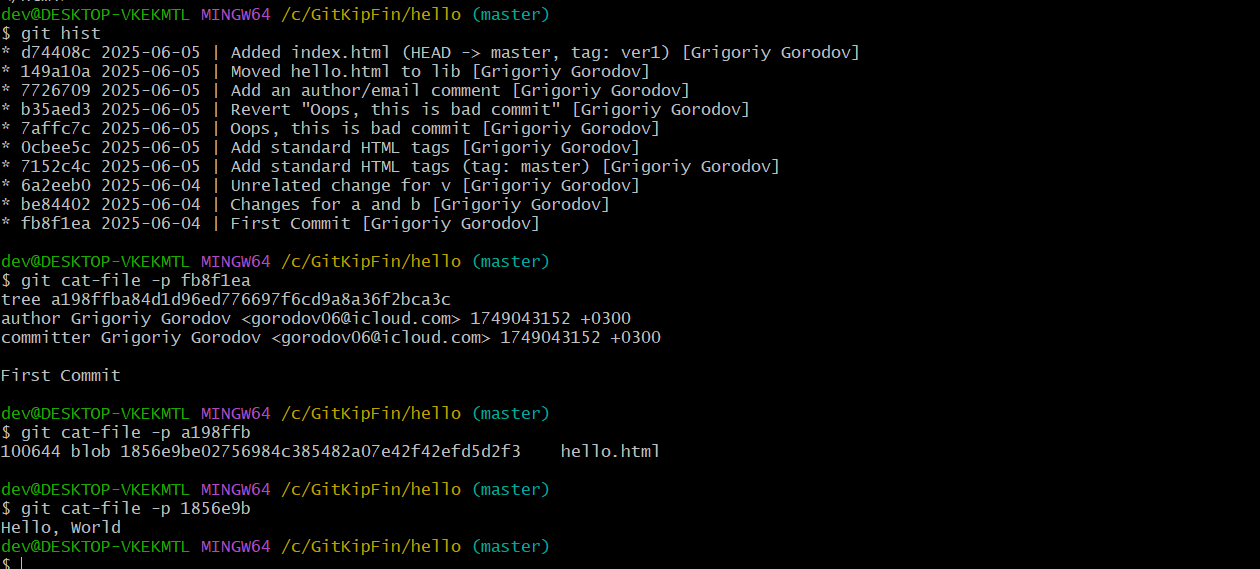


Рисунок 85 - Просмотр содержимого файла при первом коммите

# Работа с Git

## Создание ветки

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

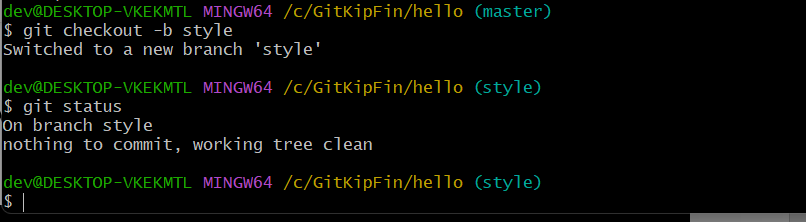


Рисунок - Создание новой ветки style

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

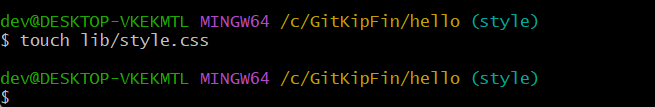


Рисунок - Создание файла стилей

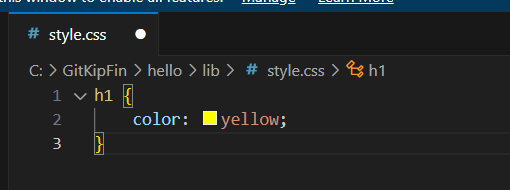


Рисунок - Код style.css

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

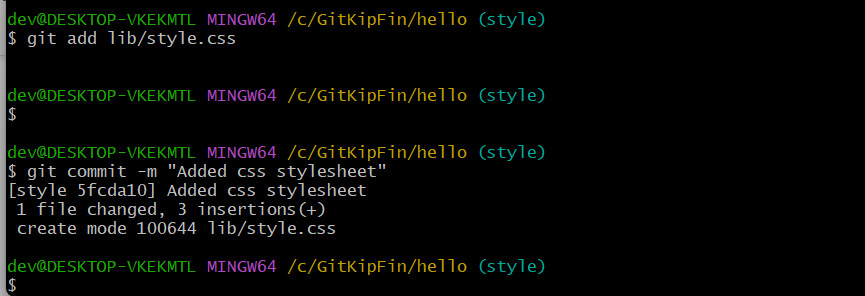


Рисунок - Индексация и коммит нового файла

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



Рисунок - Изменения в файле hello.html

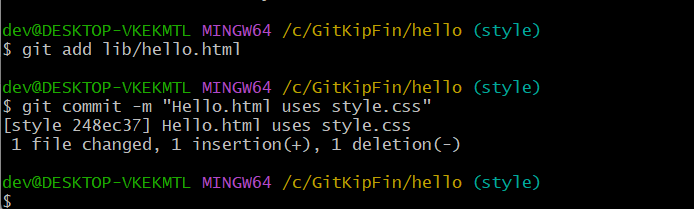


Рисунок - Индексация и коммит

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

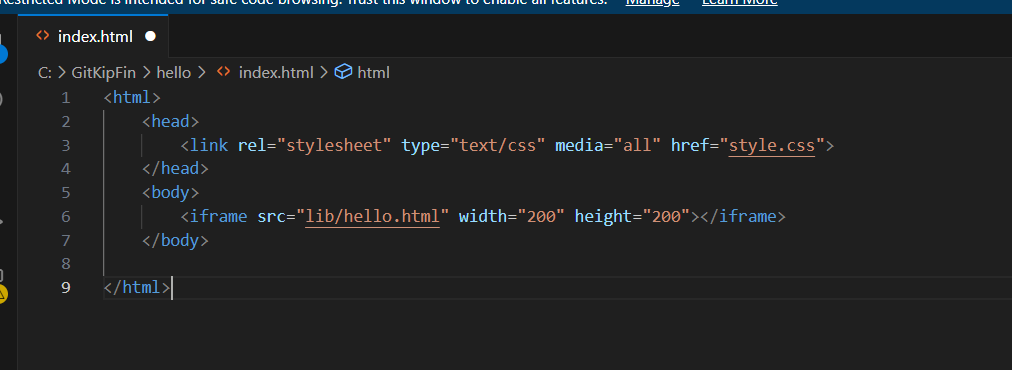


Рисунок - Изменения в файле index.html

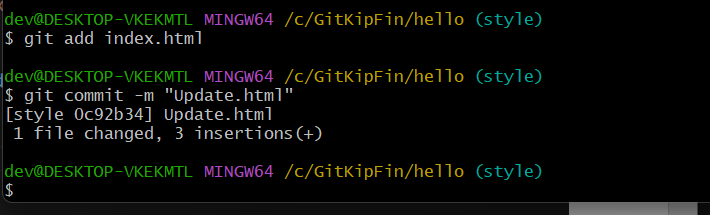


Рисунок - Индексация и коммит

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

## Навигация по веткам

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



Рисунок - Просмотр истории

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

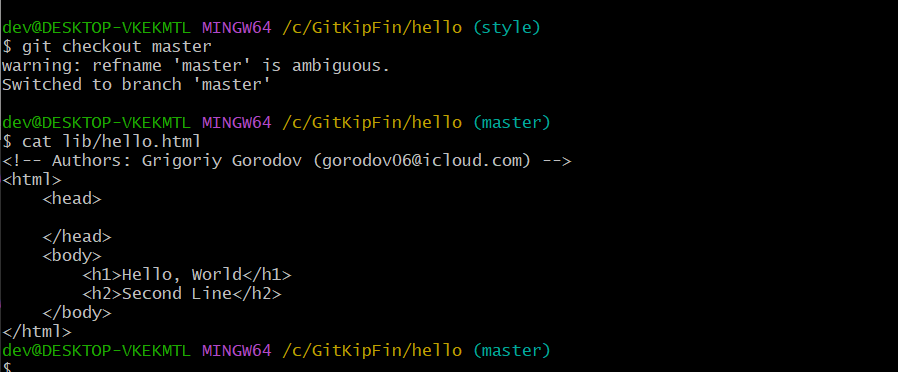


Рисунок - Переключение на ветку master

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



Рисунок - Переключение на ветку style

## Изменения в ветке master

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

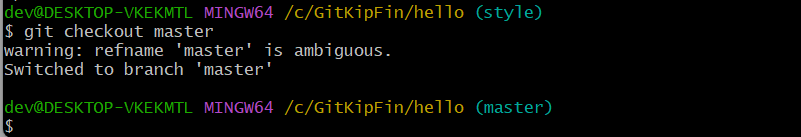


Рисунок - Переключение на ветку master

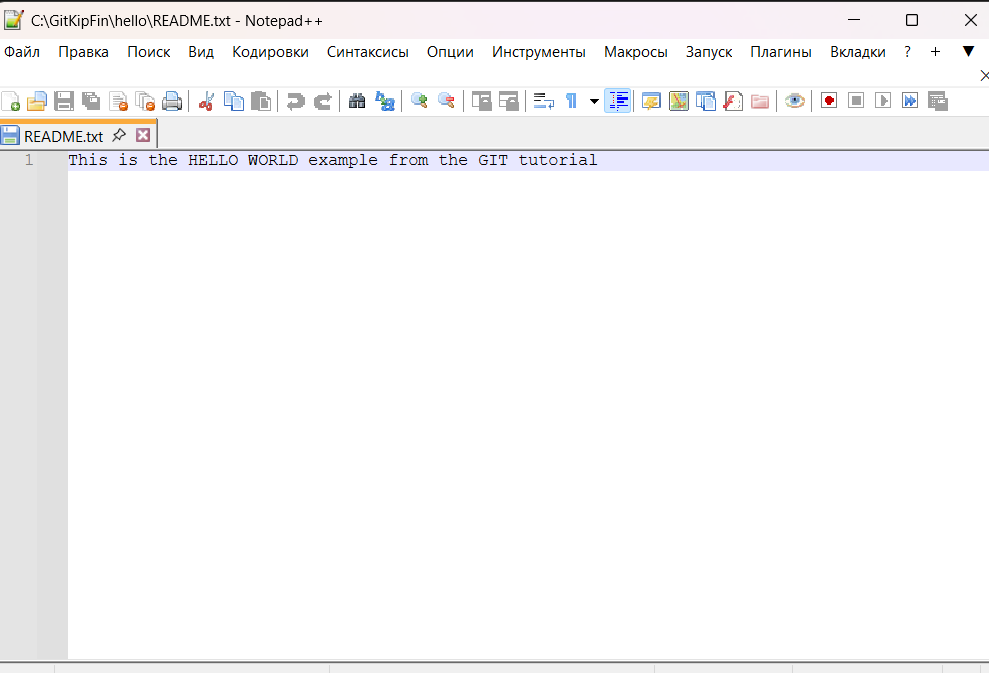


Рисунок - Содержимое файла README

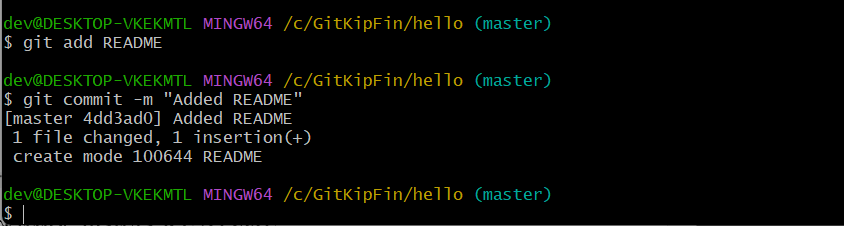


Рисунок - Индексация и коммит

## Просмотр отличающихся веток

На рисунке 15 можно увидеть дерево коммитов.



Рисунок - Дерево коммитов

## Слияние

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

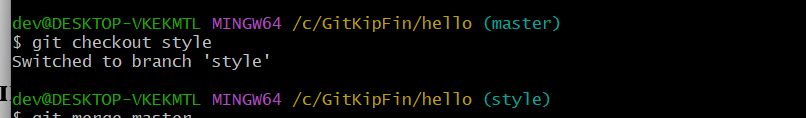


Рисунок - Переключение на ветку style

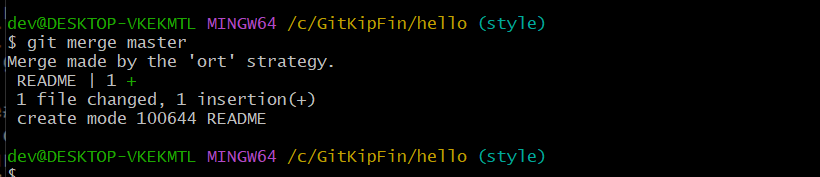


Рисунок - Слияние с веткой master



Рисунок - Просмотр истории

## Создание конфликта

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

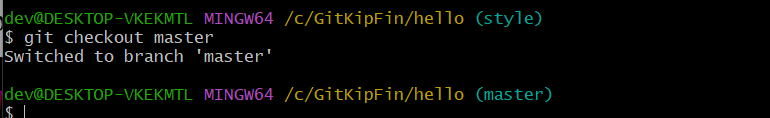


Рисунок - Переход в ветку master

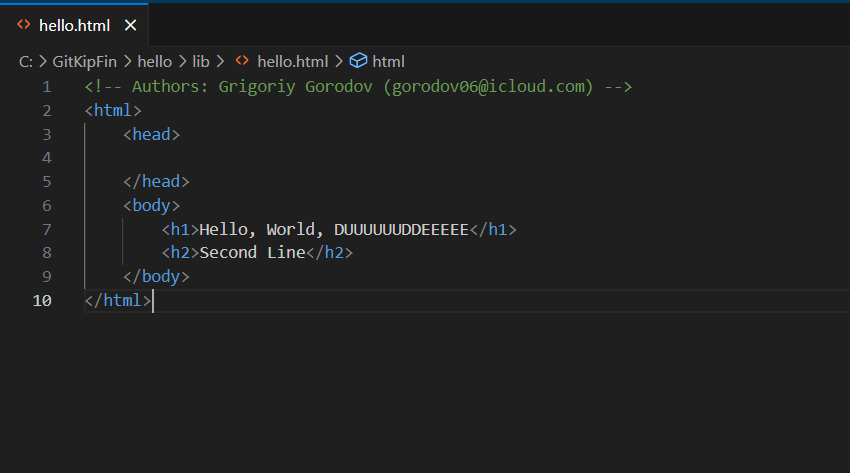


Рисунок - Внесение изменений

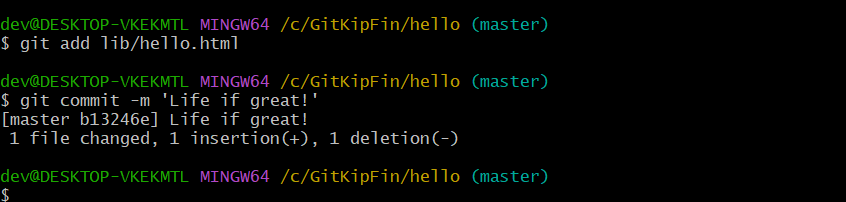


Рисунок - Индексация и коммит

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



Рисунок - Конфликт изменений

## Разрешение конфликтов

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

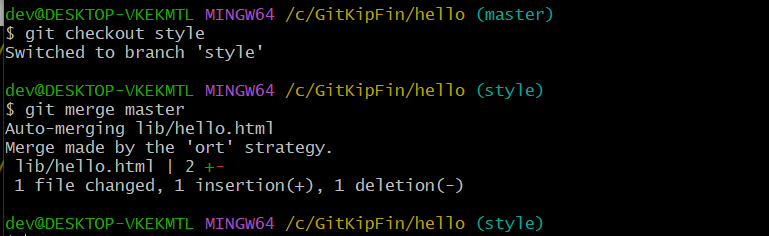


Рисунок - Ошибка при слиянии

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

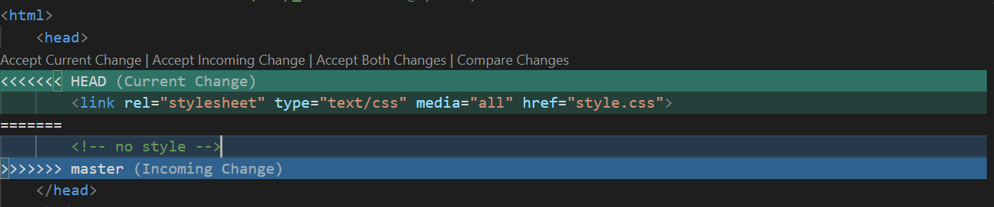


Рисунок - Просмотр файла hello.html при наличии конфликта

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

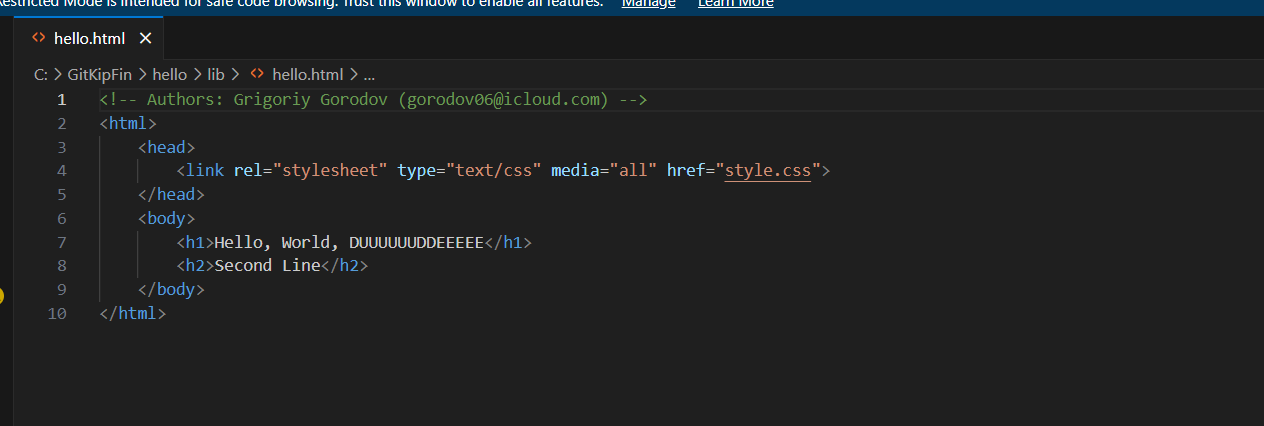


Рисунок - Решение конфликта вручную

Затем следует произвести индексацию и коммит (рисунок 26).

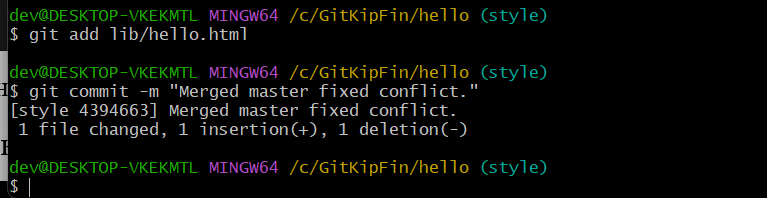


Рисунок - Индексация и коммит

# Работа с Git

## Сброс ветки style

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

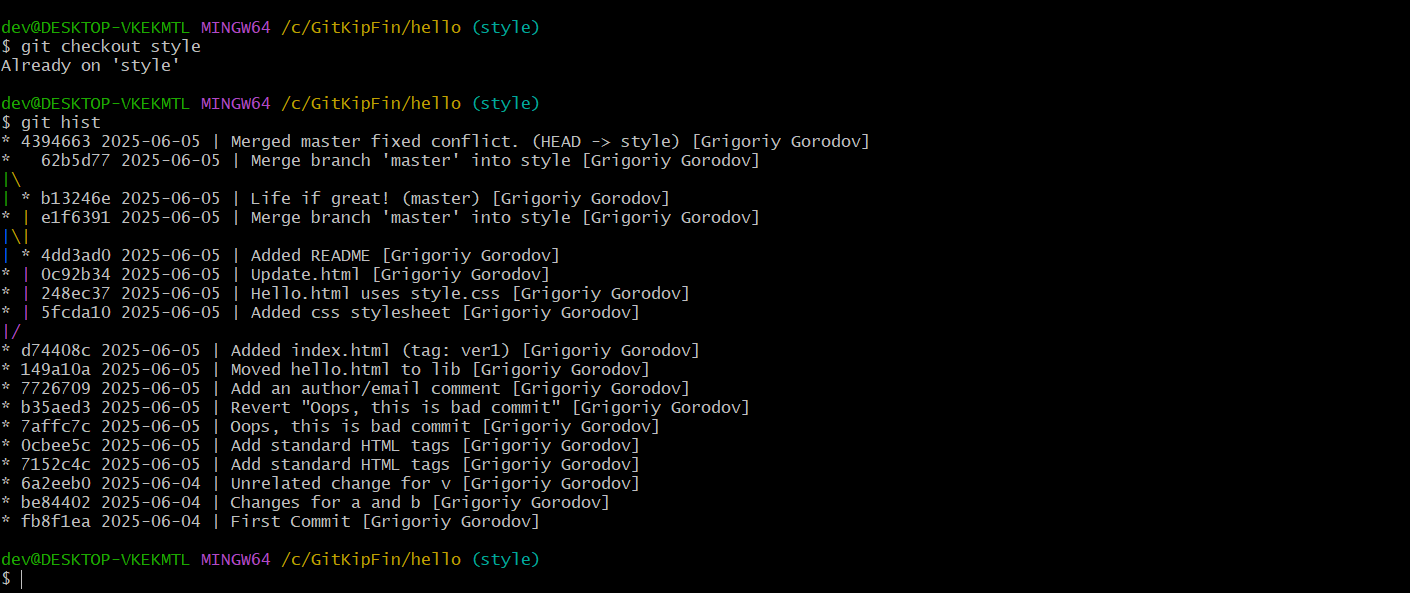


Рисунок - Просмотр истории

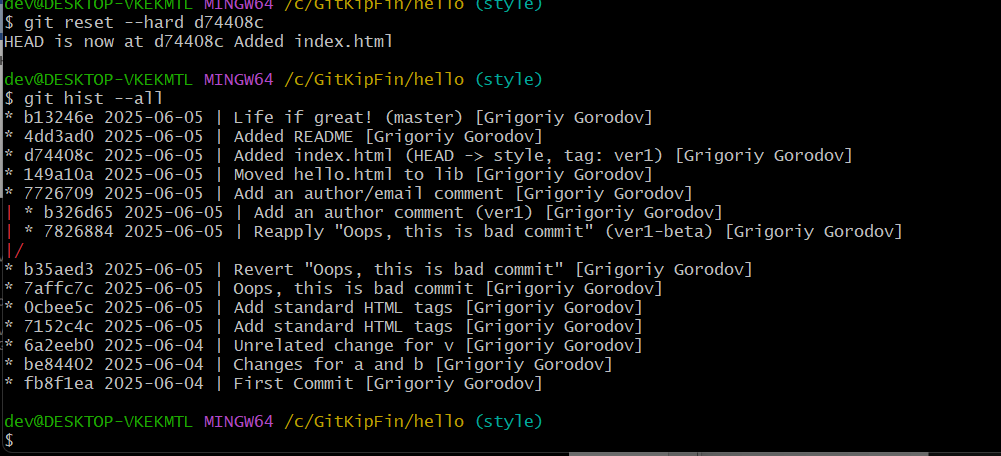


Рисунок - Сброс ветки style

## Сброс ветки master

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



Рисунок - Переключение на master и просмотр истории

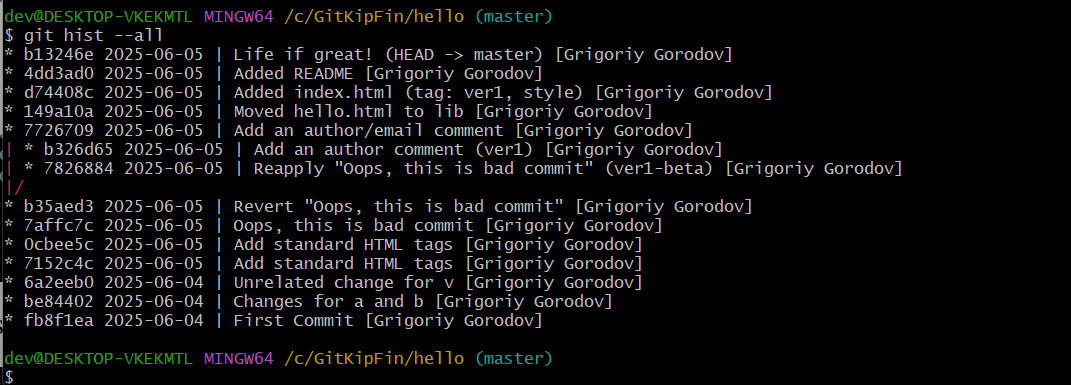


Рисунок - Сброс ветки master

## Перебазирование

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

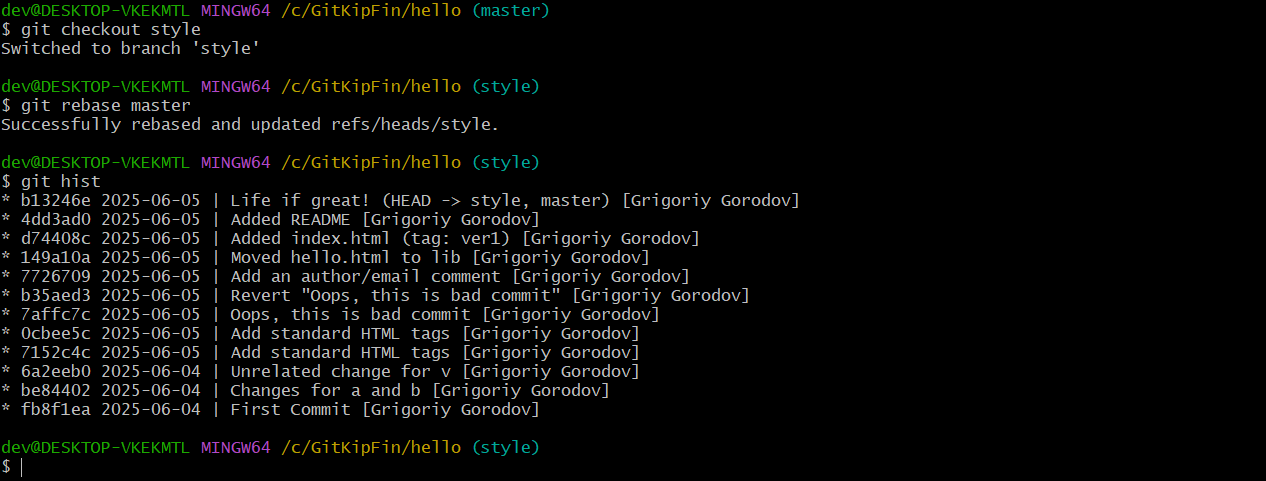


Рисунок - Перебазирование веток

## Слияние в ветку master

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

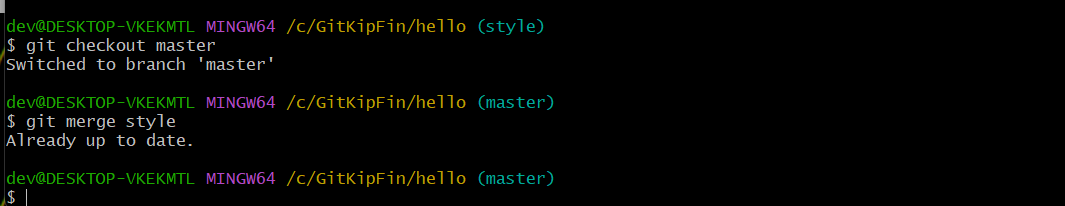


Рисунок - Слияние веток



Рисунок - Просмотр истории

## Клонирование репозиториев

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

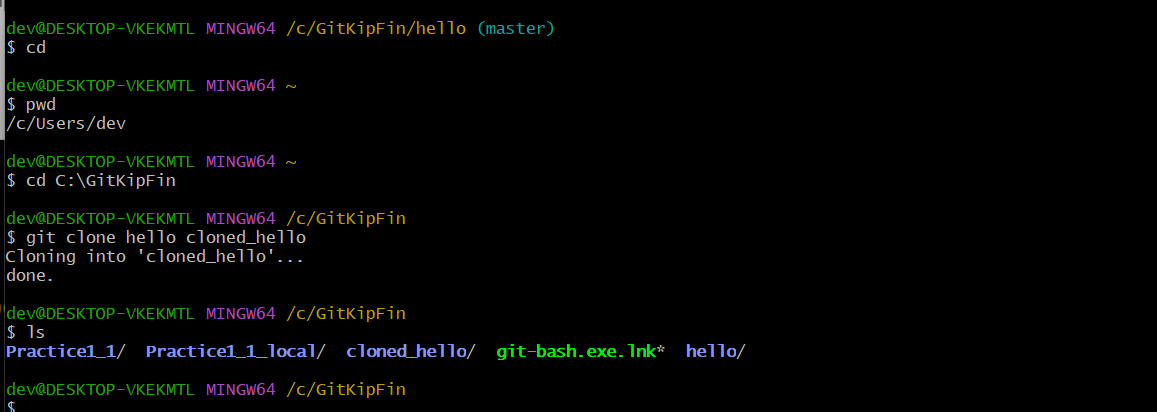


Рисунок - Переход в рабочий каталог и его клонирование

## Просмотр клонированного репозитория

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

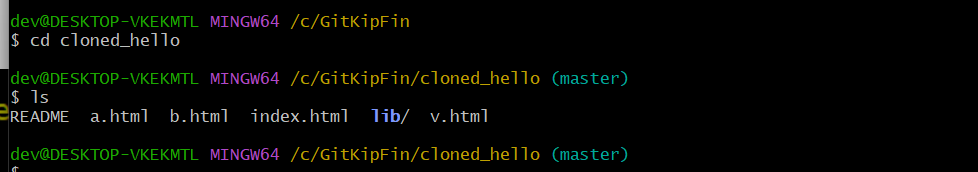


Рисунок - Просмотр содержимого клонированного репозитория



Рисунок - Просмотр логов клонированного каталога

## Origin

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

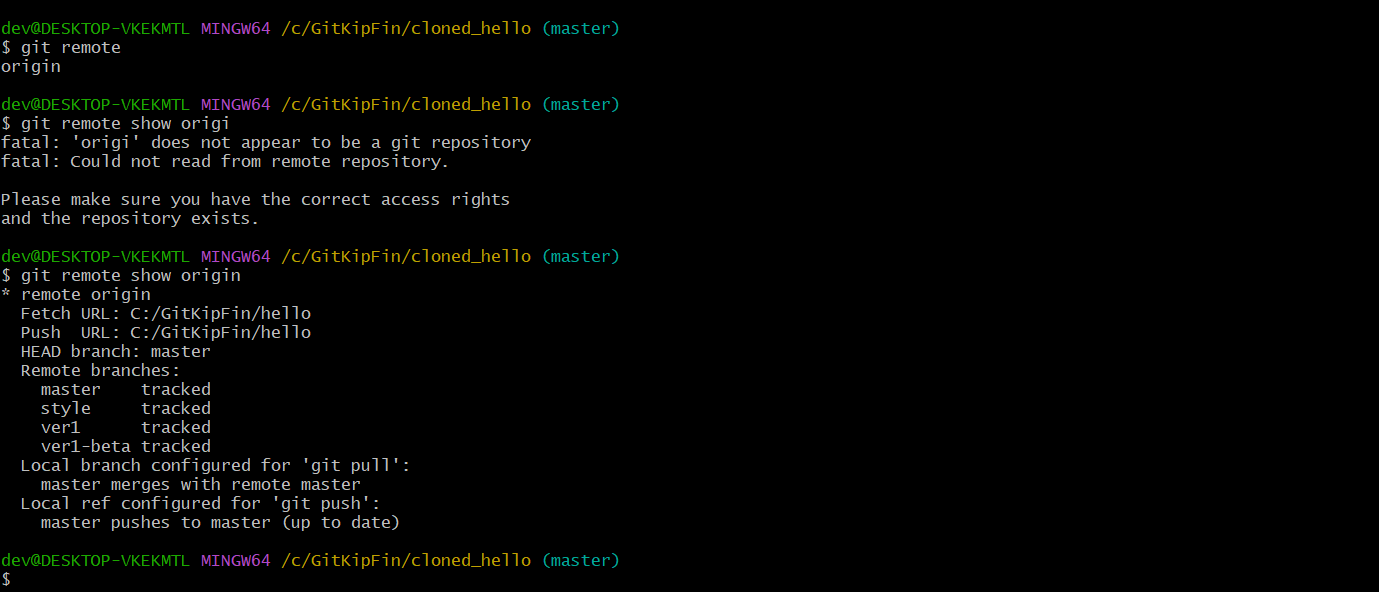


Рисунок - Просмотр данных об origin

## Удаленные ветки

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

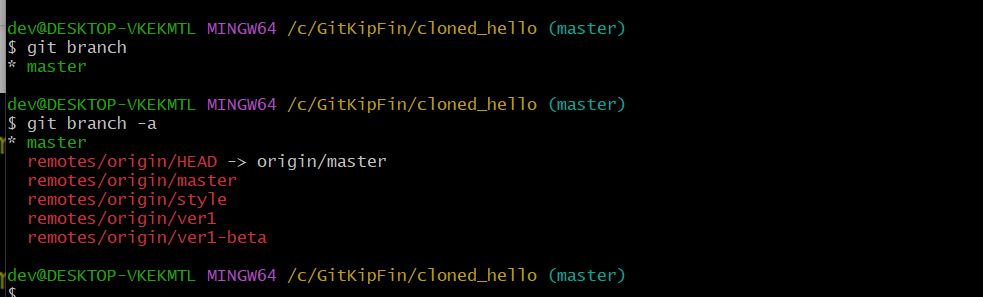


Рисунок - Просмотр удаленных веток

# Работа с Git

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

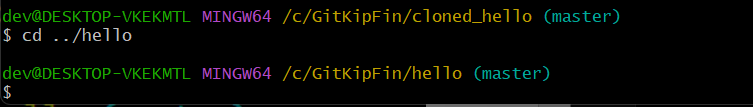


Рисунок - Переход в оригинальный репозиторий

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

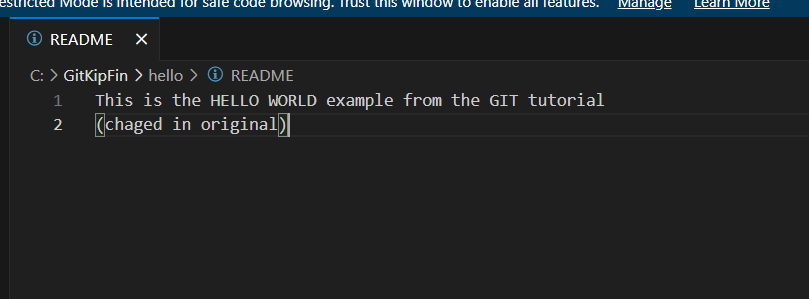


Рисунок - Изменения в файле README

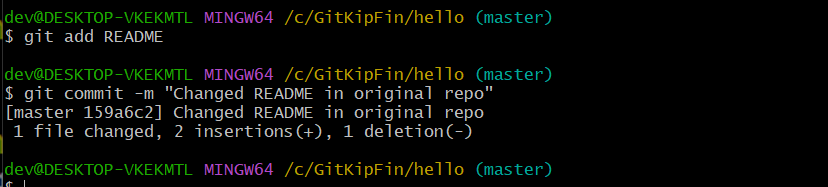


Рисунок - Индексация и коммит новых изменений

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

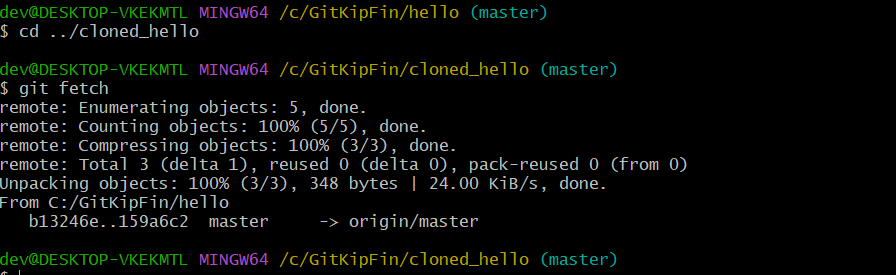


Рисунок - Извлечение изменений

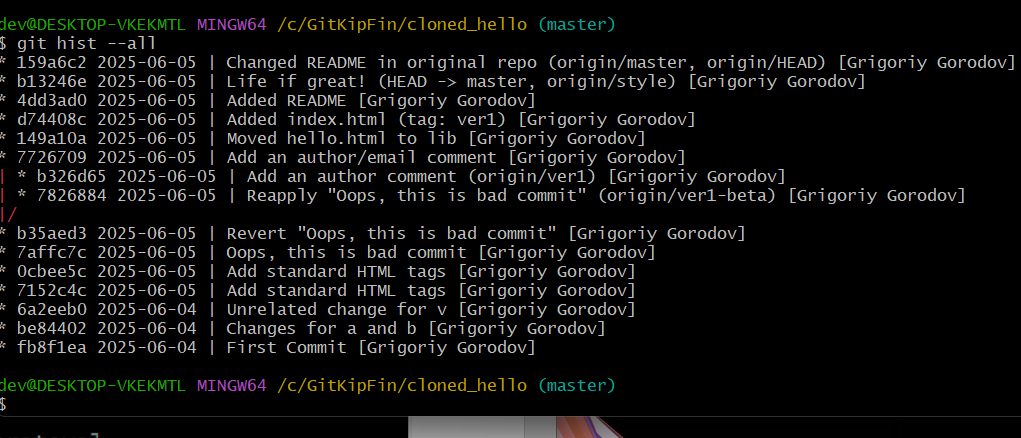


Рисунок - Просмотр истории

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

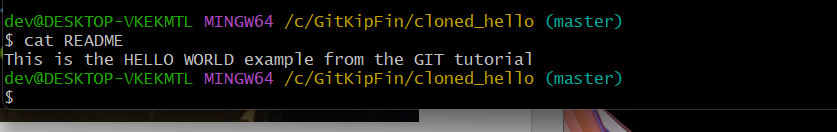


Рисунок - Вывод содержимого файла README

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

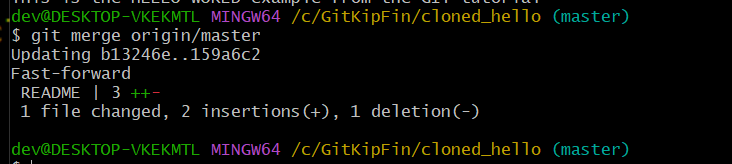


Рисунок - Слияние изменений

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

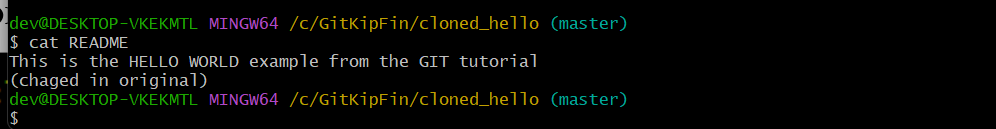


Рисунок - Вывод содержимого файла README

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

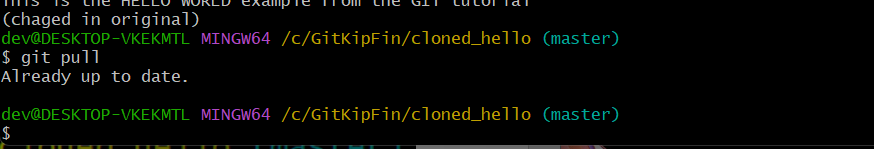


Рисунок - Команда git pull

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

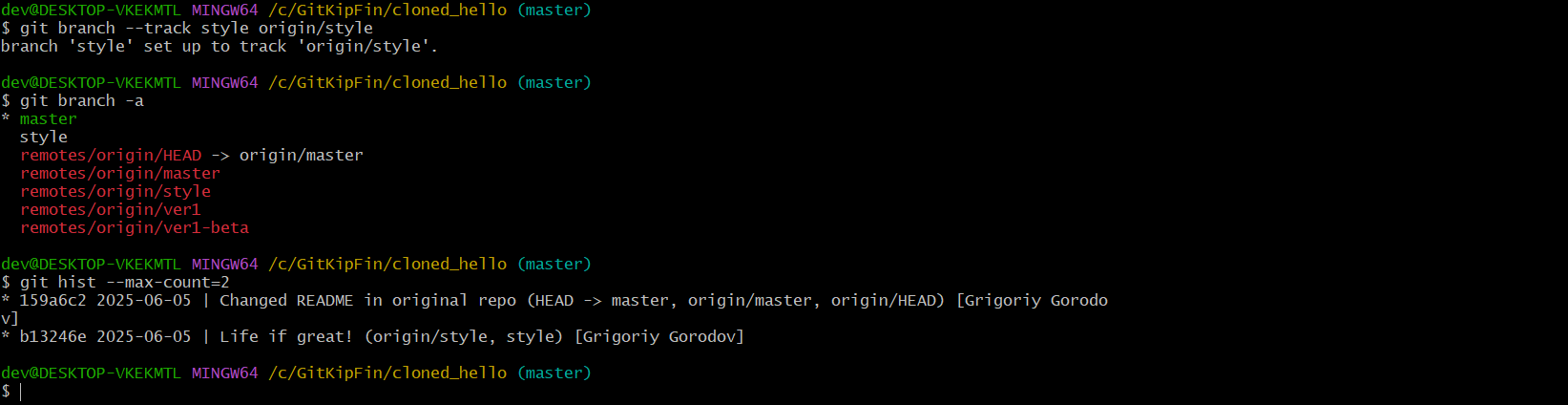


Рисунок - Добавление локальной ветки

Далее необходимо создать чистый репозиторий (рисунок 11).



Рисунок - Создание чистого репозитория

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

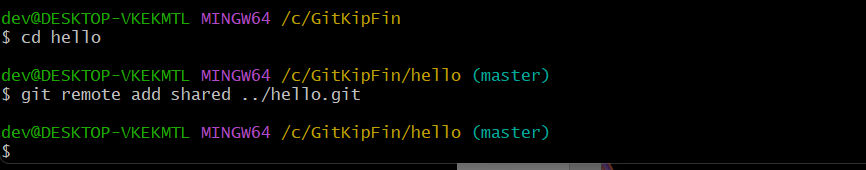


Рисунок - Добавление удаленного репозитория

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

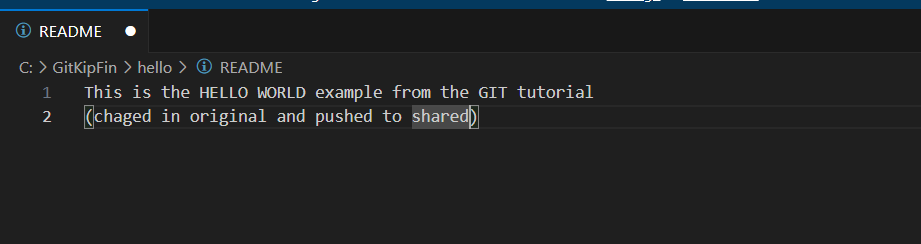


Рисунок - Внесение изменений в файл

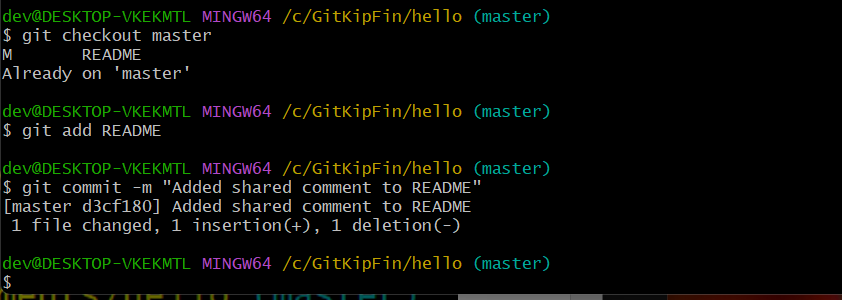


Рисунок - Индексация и коммит

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

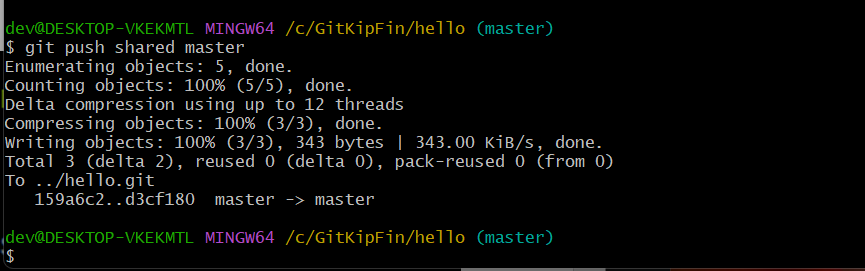


Рисунок - Отправка изменений в общий репозиторий

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

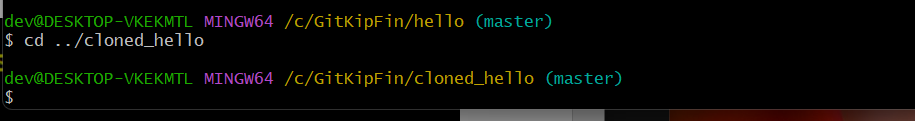


Рисунок - Переход в клонированный репозиторий

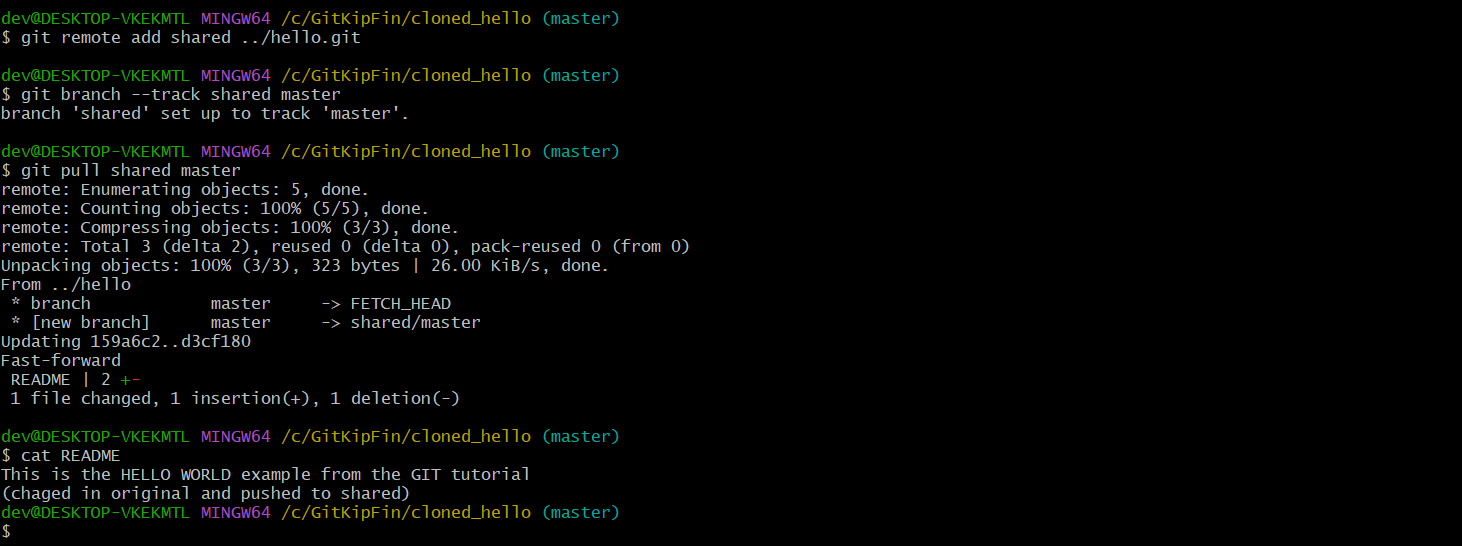


Рисунок - Команды, извлекающие общие изменения

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

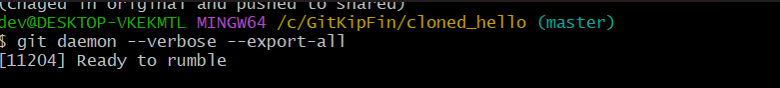


Рисунок - Настройка сервера



Рисунок - Клонирование проекта

# Работа с Git

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

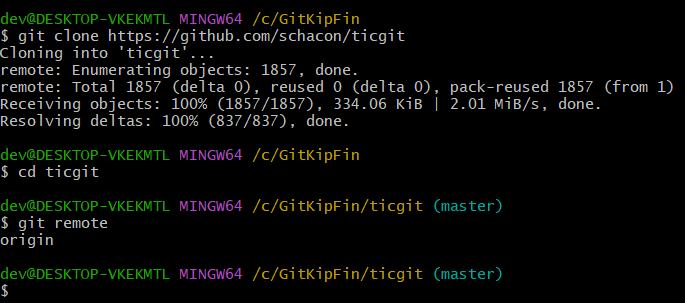


Рисунок – Клонирование репозитория и просмотр удаленных репозиториев

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

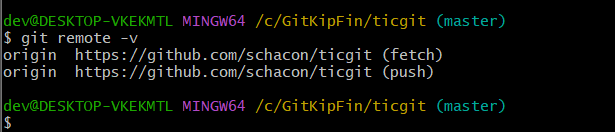


Рисунок - Просмотр удаленных репозиториев с ключом -v

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

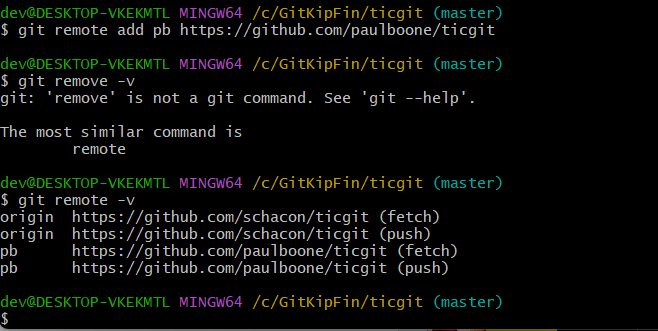


Рисунок - Добавление удаленного репозитория

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

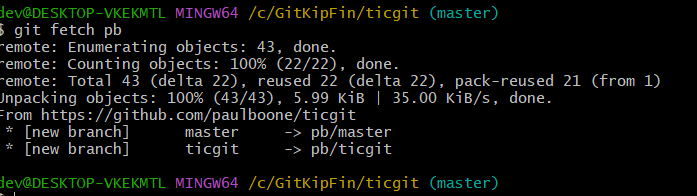


Рисунок - Использование имени вместо пути

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

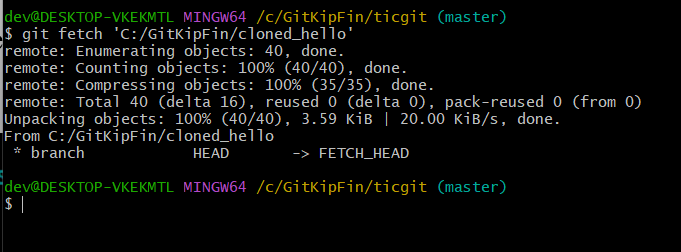


Рисунок - Получение данных из удаленных проектов

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

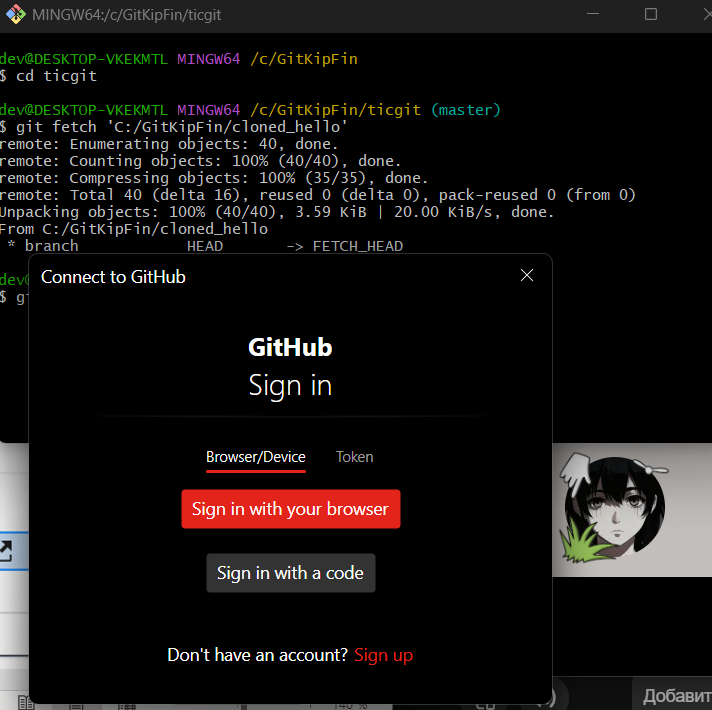


Рисунок - Отправка изменений в удаленный репозиторий

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



Рисунок - Информация об удаленном репозитории

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

Рисунок - Переименование удаленного репозитория

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

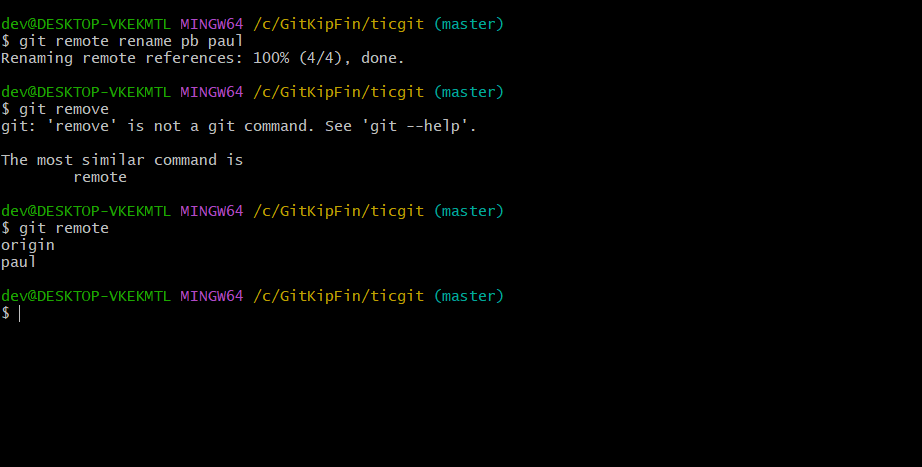


Рисунок - Удаление удаленного репозитория

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

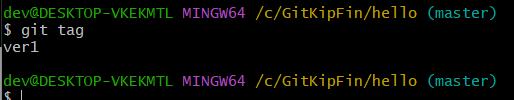


Рисунок - Просмотр тегов

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

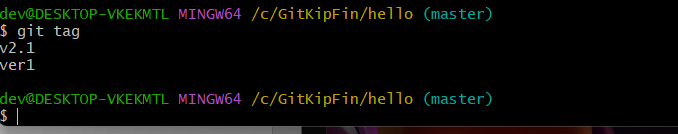


Рисунок - Создание аннотированной метки

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

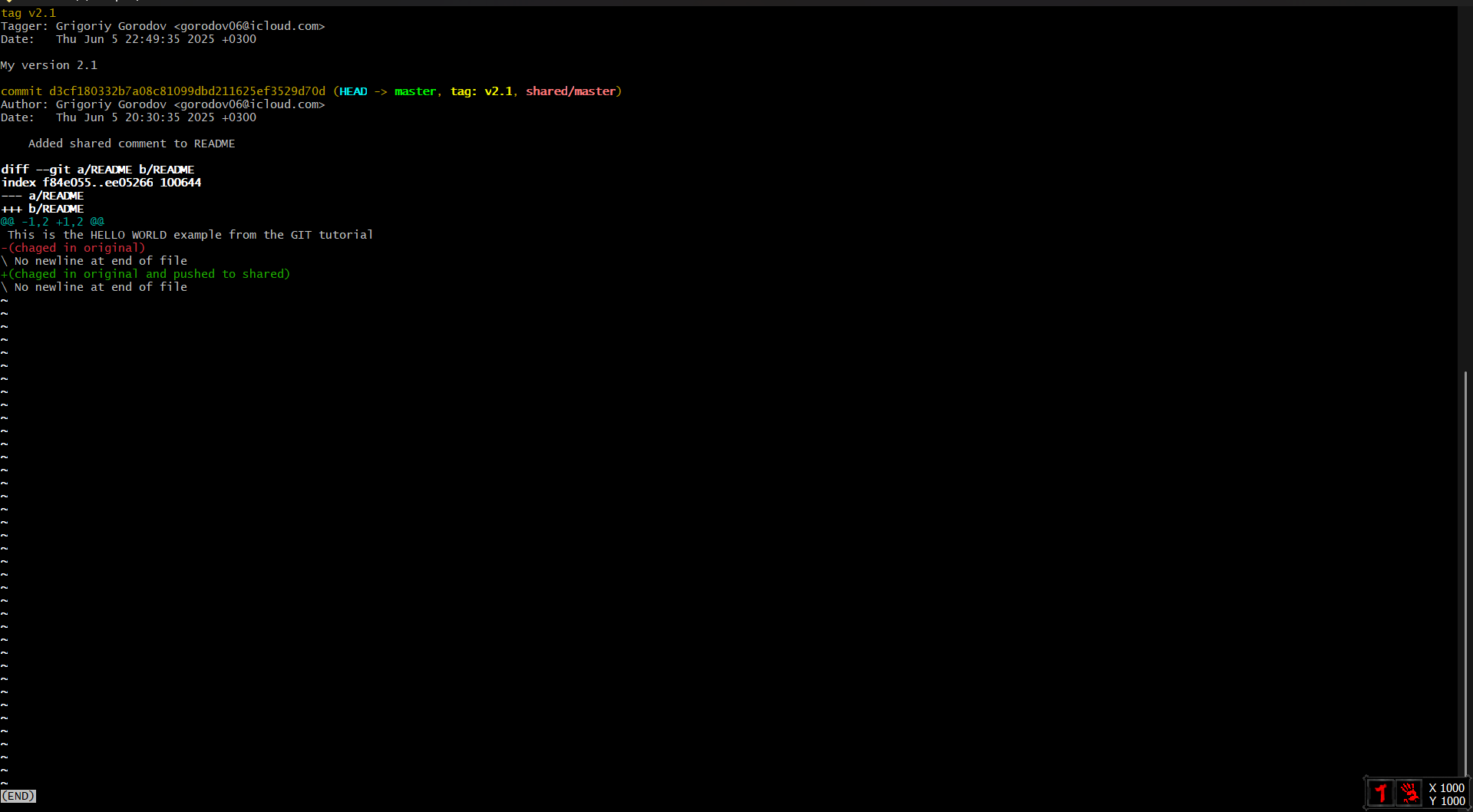


Рисунок - Просмотр данных тега

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

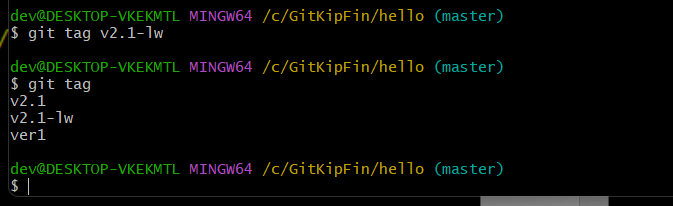


Рисунок - Создание легковесной метки

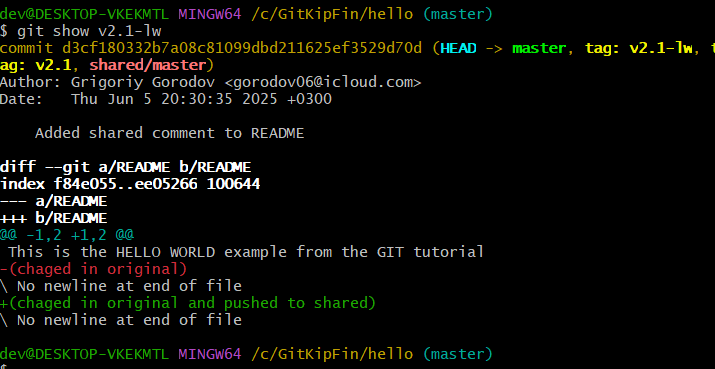


Рисунок - Просмотр данных тега

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

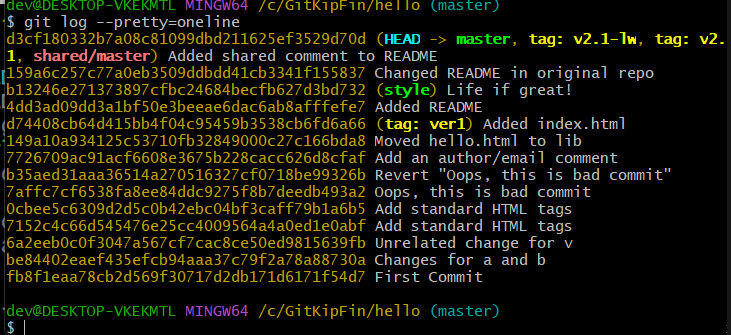


Рисунок - Просмотр истории



Рисунок - Создание тега определенному коммиту

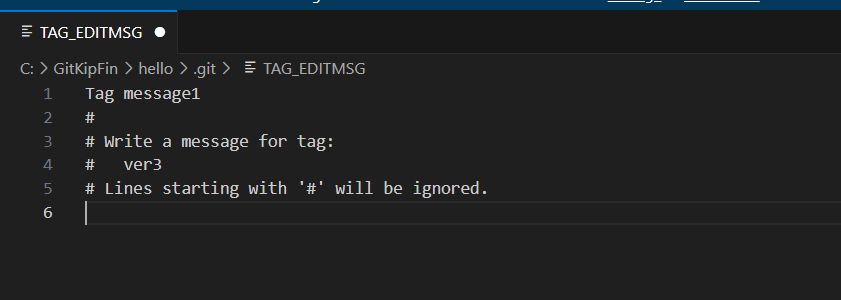


Рисунок - Ввод сообщения в текстовом редакторе

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

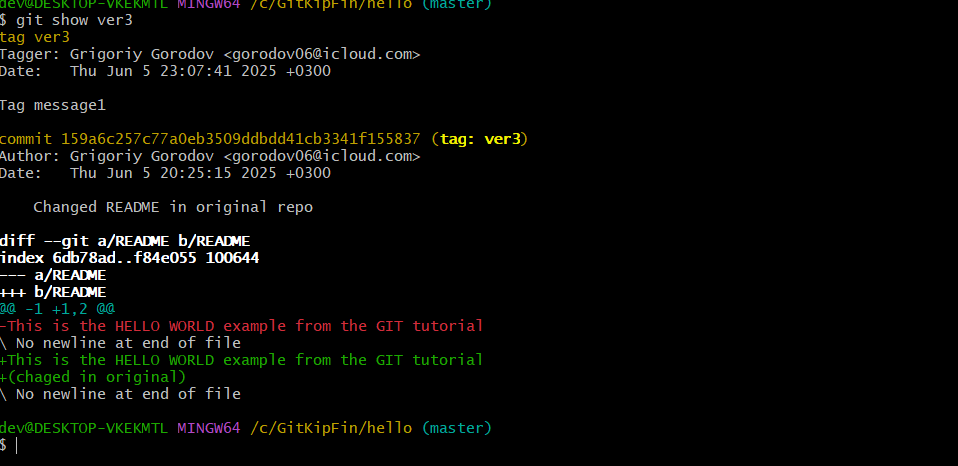


Рисунок - Просмотр данных тега

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

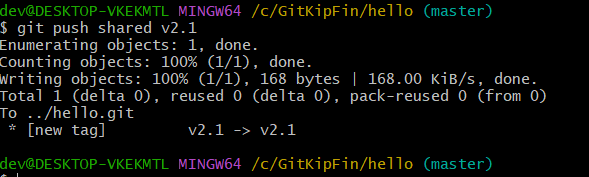


Рисунок - Отправка тега на удаленный сервер

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

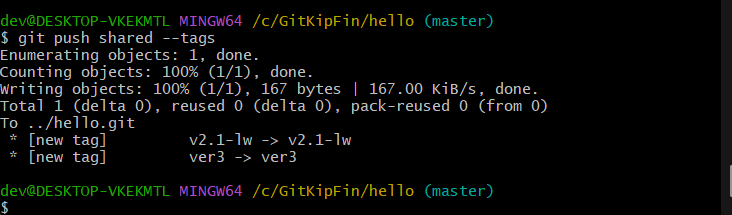


Рисунок - Отправка всех тегов на сервер

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

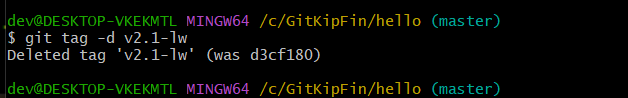


Рисунок - Удаление тега

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

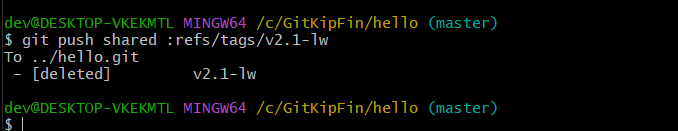


Рисунок - Удаление тегов с сервера

Для того, чтобы получить версии файлов, на которые указывает тег, можно выполнить git checkout для тега. Однако, это переведёт репозиторий в состояние «detached HEAD». Если в состоянии «detached HEAD» внести изменения и сделать коммит, то тег не изменится, при этом новый коммит не будет относиться ни к какой из веток, а доступ к нему можно будет получить только по его хэшу. Поэтому в таком случае следует создать новую ветку (рисунки 23-24).

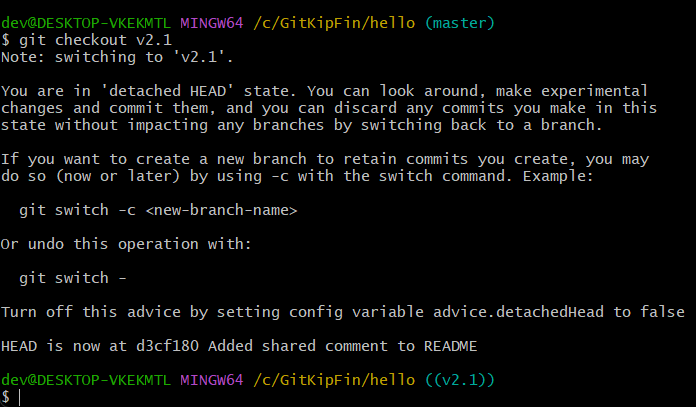


Рисунок - Переключение на метку

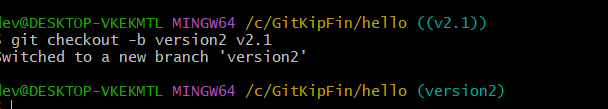


Рисунок - Создание новой ветки

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

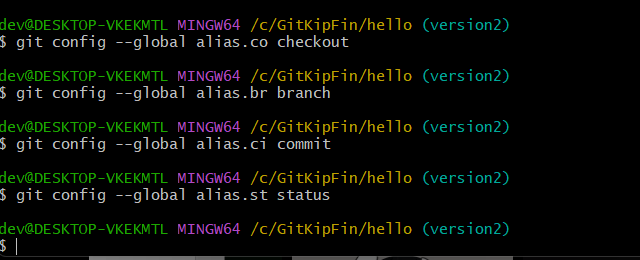


Рисунок - Задание алиасов

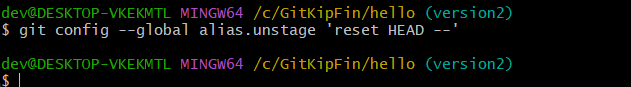


Рисунок - Создание псевдонима исключения файла из индекса

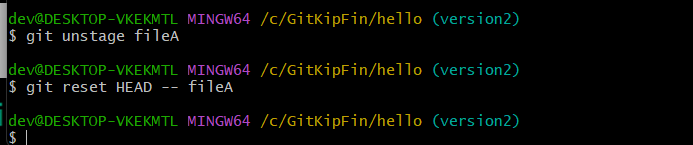


Рисунок - Использование созданного псевдонима



Рисунок - Создание алиаса для просмотра последнего коммита

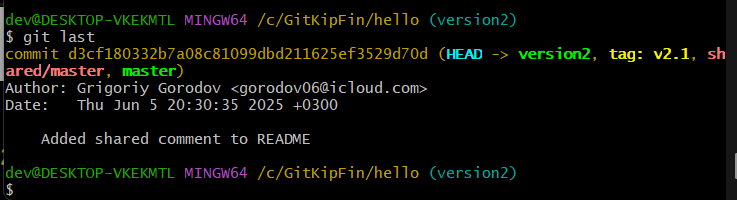


Рисунок - Результат работы созданного алиаса

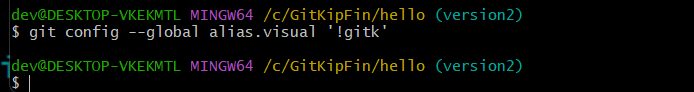


Рисунок - Создание псевдонима внешней команды

# Работа с Git

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

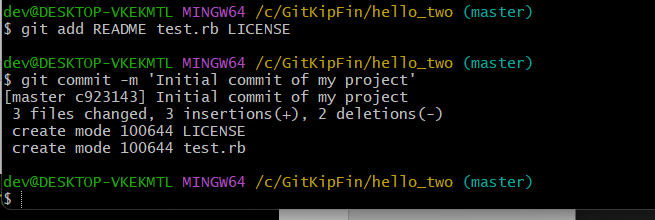


Рисунок 1 - Индексация и коммит 3 файлов

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

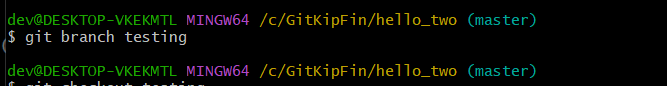


Рисунок 2 - Создание ветки testing

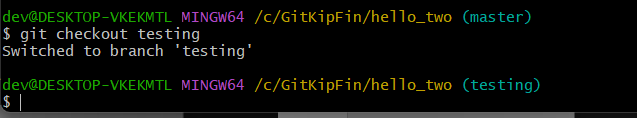


Рисунок 3 - Переключение на ветку testing

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

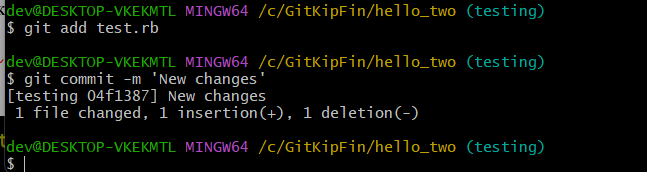


Рисунок 4 - Индексация и коммит файла test.rb

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



Рисунок 5 - Переключение на ветку master

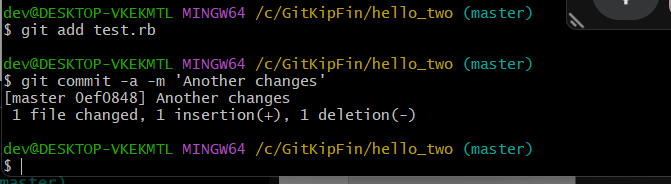


Рисунок 6 - Еще индексация и коммит test.rb

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

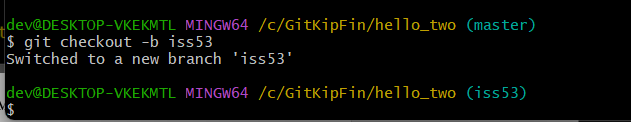


Рисунок 7 - Создание и переключение на ветку iss53

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

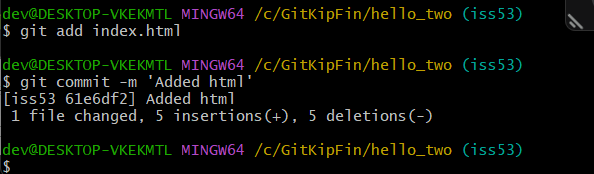


Рисунок 8 - Индексация и коммит файла index.html

Далее нужно переключить ветку на master (рисунок 9).

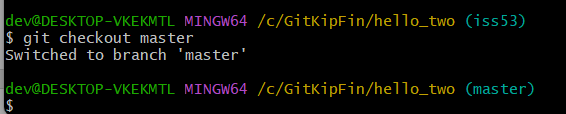


Рисунок 9 - Переключение на ветку master

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

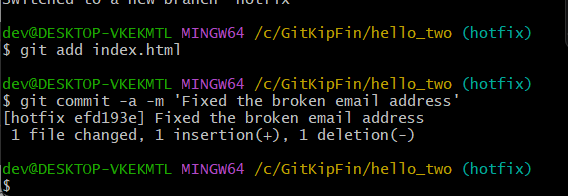


Рисунок 10 - Индексация и коммит index.html на ветке hotfix

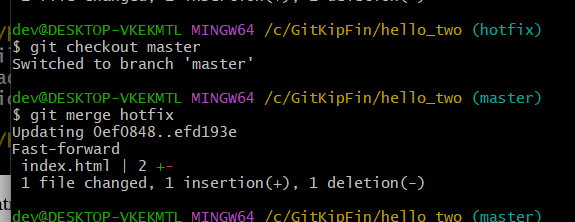


Рисунок 11 - Переключение на ветку master и объединение с веткой hotfix

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

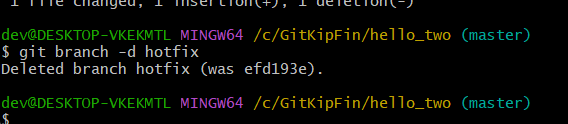


Рисунок 12 - Удаление ветки hotfix

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

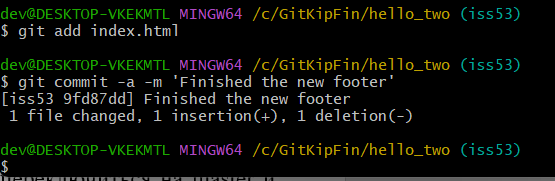


Рисунок 13 - Индексация и коммит index.html на ветке iss53

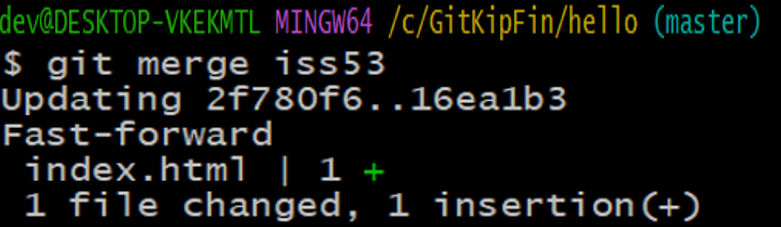


Рисунок 14 - Слияние веток

После этого ветку iss53 нужно удалить (рисунок 15).

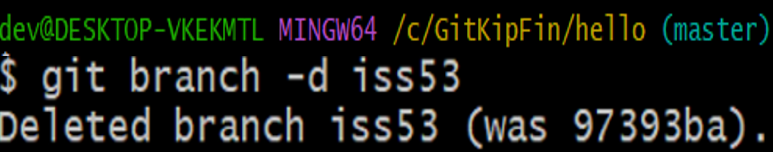


Рисунок 15 - Удаление ветки iss53

# Работа с Git

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

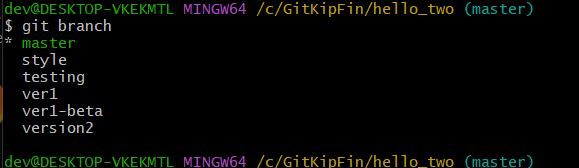


Рисунок 116 - Список существующих веток

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

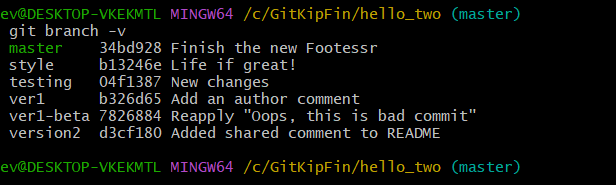


Рисунок 117 - Список веток с последними коммитами

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

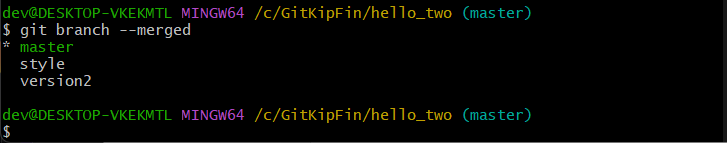


Рисунок 118 - Список веток слитых с текущей

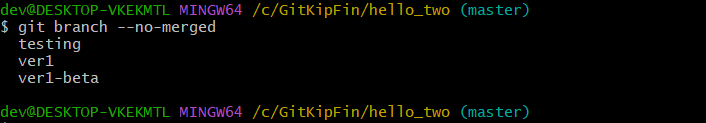


Рисунок 119 - Список веток не слитых с текущей

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

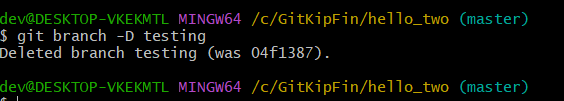


Рисунок 120 - Удаление ветки

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

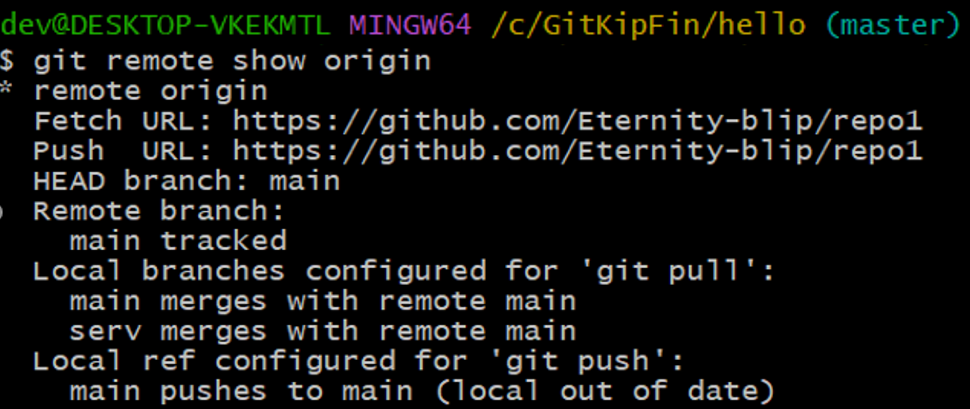


Рисунок 121 - Просмотр удаленных веток

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



Рисунок 122 - Отправка изменений

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

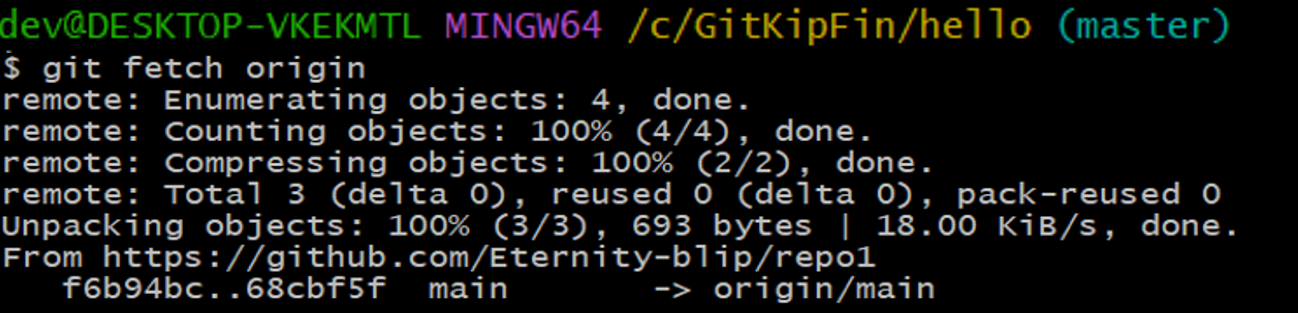


Рисунок 123 - Выполнение команды git fetch

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



Рисунок 124 - Создание ветки на основе удаленной ветки

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



Рисунок 125 - Удаление ветки на сервере

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

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

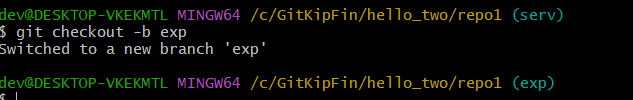


Рисунок 126 - Перемещение изменений

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

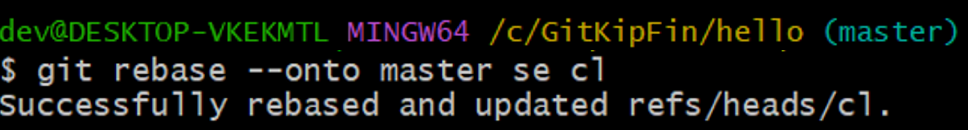


Рисунок 127 - Перемещение изменений с параметром onto

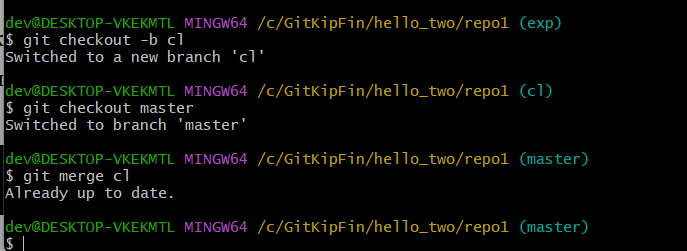


Рисунок 128 - Слияние веток master и cl

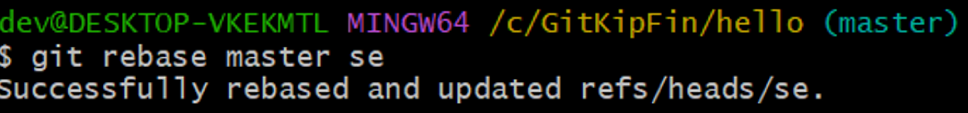


Рисунок 129 - Перемещение изменений

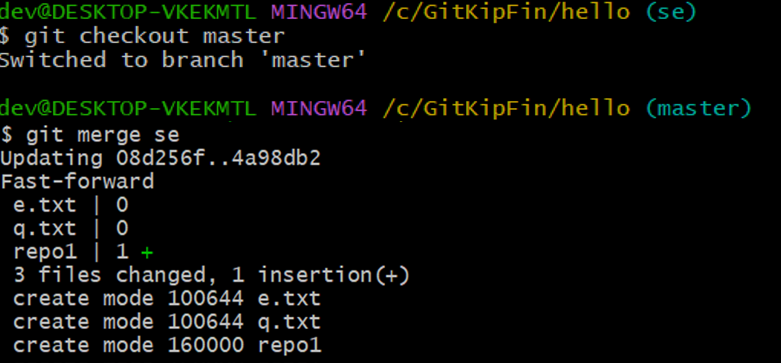


Рисунок 130 - Слияние веток master и se

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

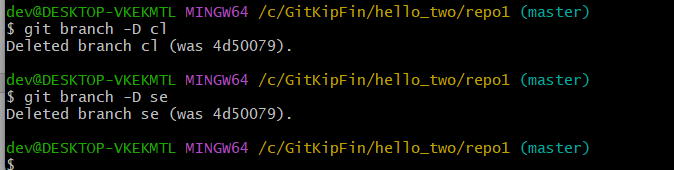


Рисунок 131 - Удаление веток cl и se