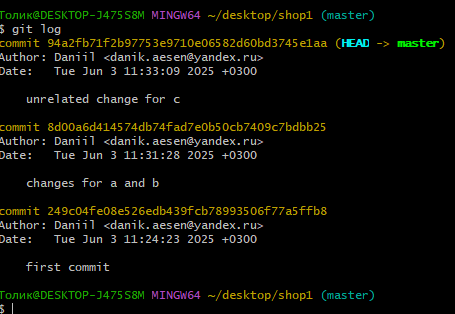
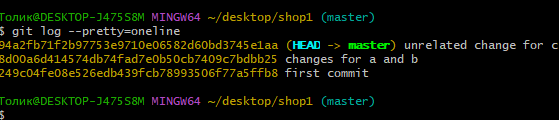
\_УП\_2

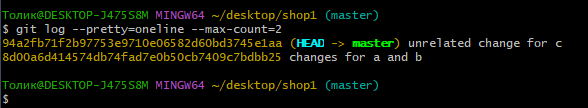
Для того, чтобы просмотреть список произведенных изменений в проекте, используется команда git log (рисунок 1).

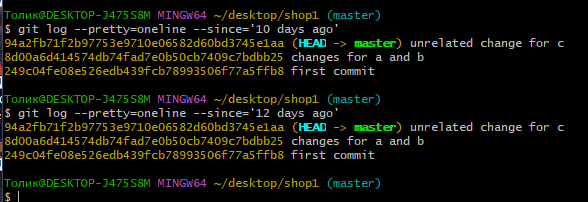


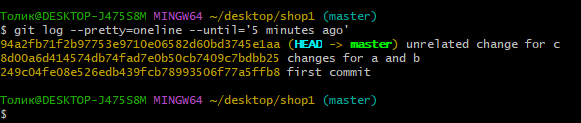
На рисунке 1 была выведена полная история. Для того, чтобы увидеть однострочный формат используется команда git log –pretty=oneline (рисунок 2).

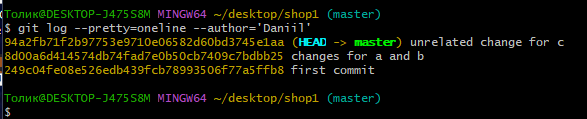


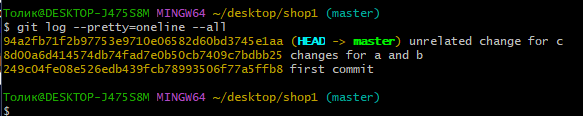
Далее на рисунках 3-8 показано несколько вариантов вывода истории изменений.

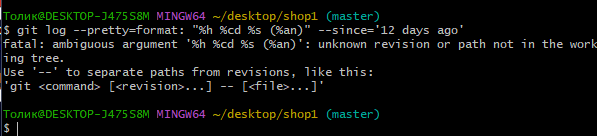


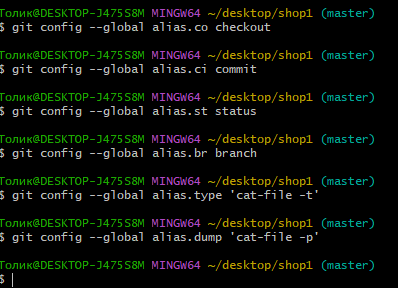


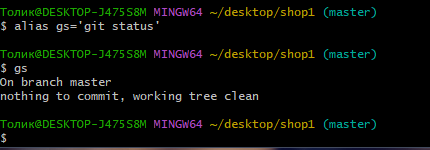




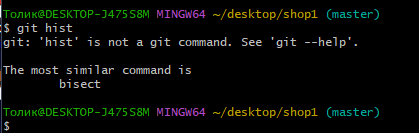




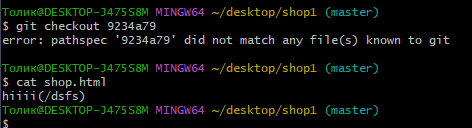




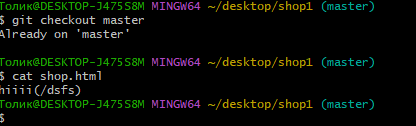
Для того, чтобы вернуть рабочий каталог к предыдущему состоянию, можно использовать следующий способ: для начала нужно узнать хэши предыдущих версий, что можно сделать с помощью ранее заданного алиаса hist (рисунок 12).



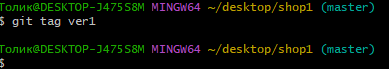
Далее нужно выполнить команду git checkout с номером нужного хэша (достаточно первых 7 знаков). После этого можно просмотреть содержимое файла с помощью команды cat (рисунок 13).



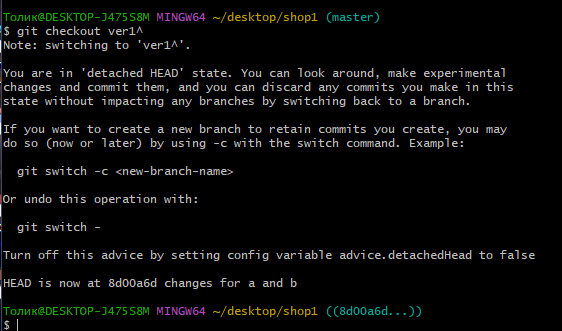
Для возвращения к последней версии в ветке master (имя ветки по умолчанию) надо ввести команду git checkout master, что показано на рисунке 14.

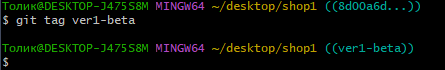


Для создания тега используется команда git tag. На рисунке 15 показано, тегом ver1 была названа текущая версия страницы.

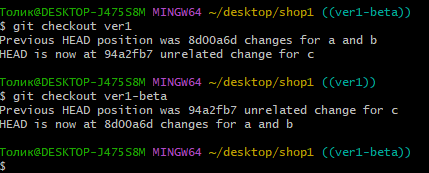


Чтобы перейти к предыдущей версии, можно использовать символ «^», который означает «родитель».

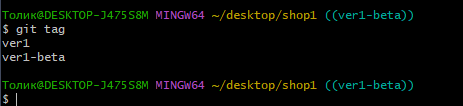




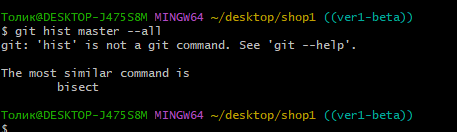
Теперь с помощью тегов можно переключаться между версиями (рисунок 18).



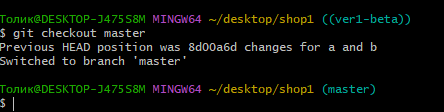
Для просмотра всех тегов используется команда git tag (рисунок 19).



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



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



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

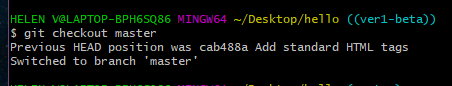


Рисунок 21 - Переключение на последний коммит

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

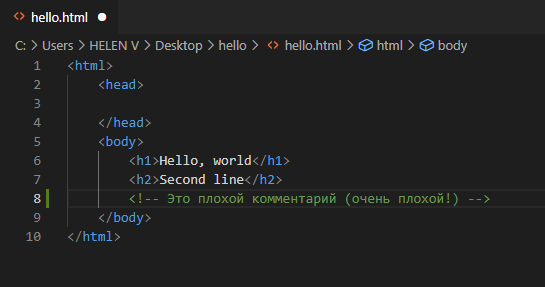


Рисунок 22 - Внесение изменения в файл

После выполнения команды git status будет показано, что есть не проиндексированное изменение (рисунок 23).

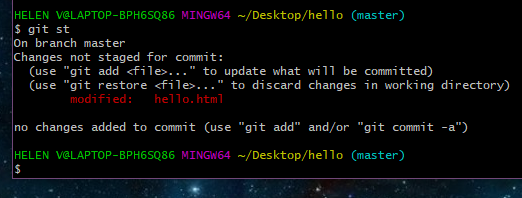


Рисунок 23 - Не проиндексированное изменение

Для переключения в версию файла без изменений используется команда git checkout hello.html (рисунок 24). Команда git status покажет, что не было произведено изменений, не зафиксированных в каталоге.

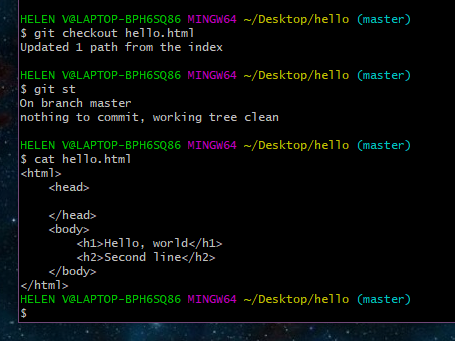


Рисунок 24 - Возвращение к версии

## Отмена проиндексированных изменений (перед коммитом)

Для того, чтобы научиться отменять проиндексированные изменения, сначала нужно внести ненужное изменение в файл (рисунок 25). После этого производится индексация (рисунок 26).

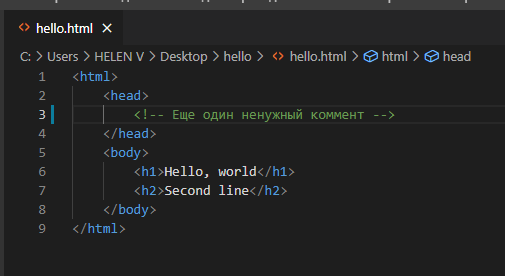


Рисунок 25 - Внесение ненужного изменения

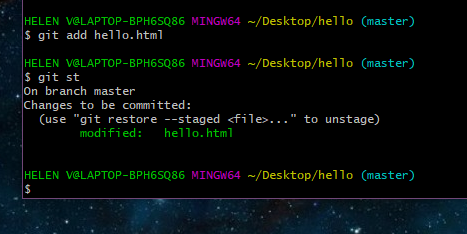


Рисунок 26 - Индексация изменения

Для отмены индексация изменения используется команда git reset HEAD hello.html (рисунок 27). Команда reset сбрасывает буферную зону к HEAD и очищает ее от проиндексированных изменений. Но для удаления ненужного по-прежнему используется команда git checkout (рисунок 28).

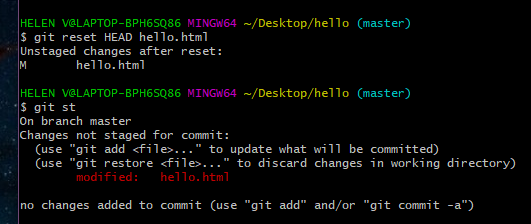


Рисунок 27 - Очистка буферной зоны

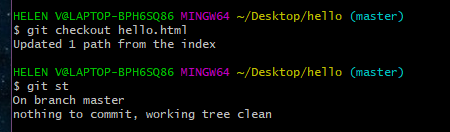


Рисунок 28 - Удаление ненужных изменений

## Отмена коммитов

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

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

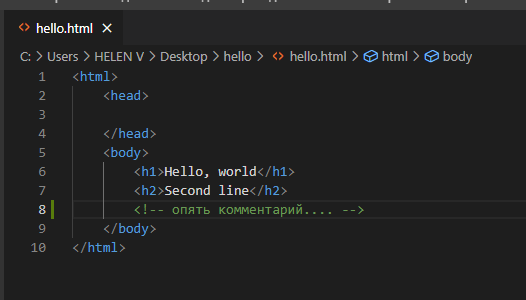


Рисунок 29 - Внесение изменения в файл

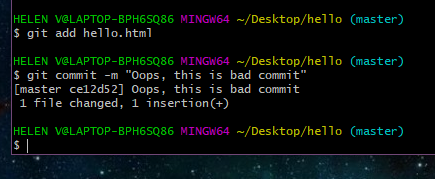


Рисунок 30 - Индексация и коммит

Для создания коммита, который удалит ненужные изменения, используется команда git revert HEAD (рисунок 31). После этого будет открыт редактор, в котором можно отредактировать коммит сообщение (рисунок 32), затем надо сохранить файл и закрыть редактор (рисунок 33).

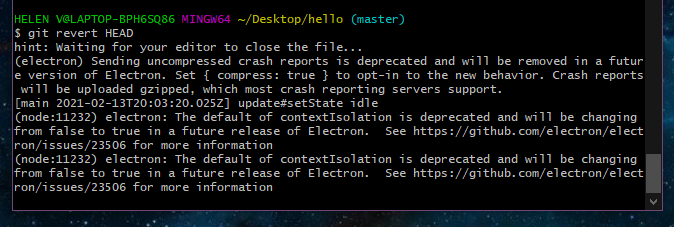


Рисунок 31 - Выполнение команды git revert

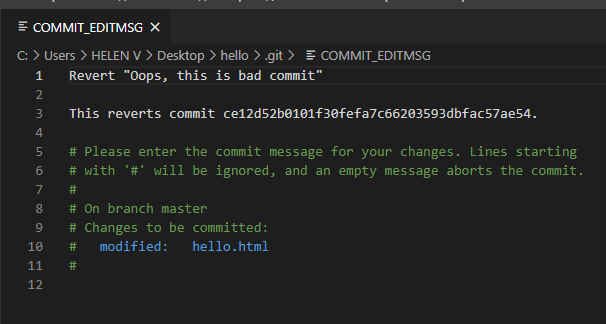


Рисунок 32 - Коммит сообщение в редакторе

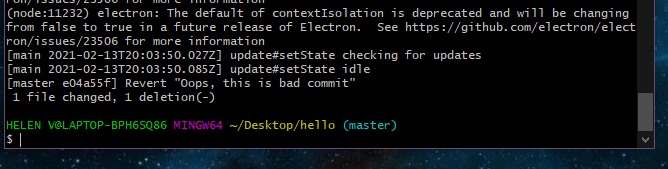


Рисунок 33 - Редактор закрыт

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

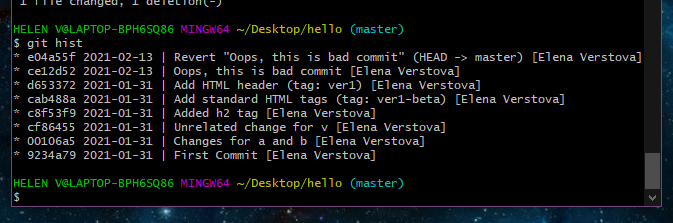


Рисунок 34 - Все коммиты при просмотре лога

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

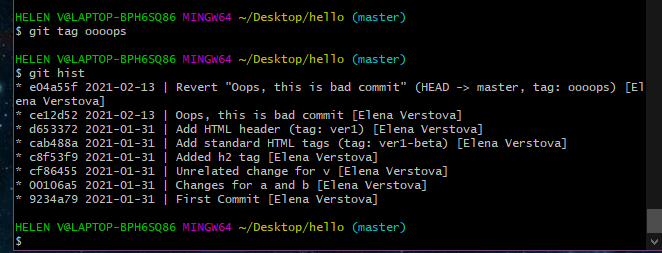


Рисунок 35 - Отметка тегом коммита

Для сброса коммитов используется команда git reset –hard ver1 (рисунок 36). Она сбрасывает ветку до версии с тегом ver1.

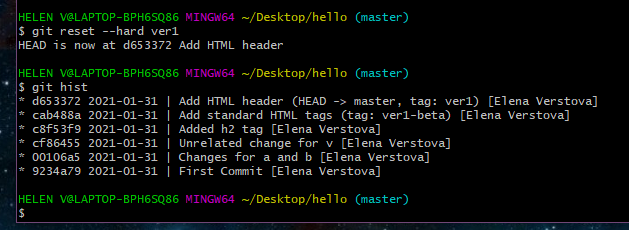


Рисунок 36 - Сброс коммита

Но при просмотре лога с помощью команды git hist –all отмененные коммиты по-прежнему будут показываться, так как они всё еще находятся в репозитории (рисунок 37).

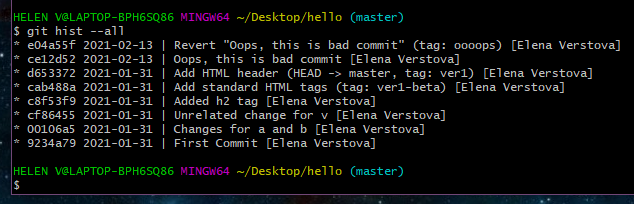


Рисунок 37 - Сброшенные коммиты находятся по-прежнему в репозитории

## Удаление тега

Так как тег «oooops» больше не нужен, его и коммиты, на которые он указывает, можно удалить с помощью команды git tag -d (рисунок 38).

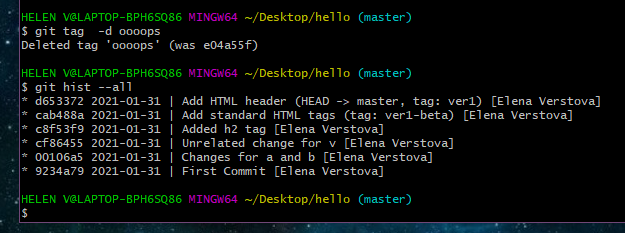


Рисунок 38 - Удаление тега