

# Система контроля версий Git

Основы продолжение

# Основные операции

- Были сделали изменения в файле test.cpp. Проверить статус:
  - `$ git status`
- Добавить изменения в **stage**:
  - `$ git add test.cpp` или `$ git add .`
- Сделать коммит в локальный репозиторий:
  - `$ git commit -m "my first commit"`
- Проверить изменения с удаленного репозитория и добавит в локальный:
  - `$ git pull`
- Добавить изменения в удаленный репозиторий:
  - `$ git push`

# Просмотр истории коммитов

- Посмотреть что было сделано — историю коммитов:
  - `$ git log`
- Показывает разницу, внесенную в каждый коммит, для 2 записей:
  - `$ git log -p -2`
- Показывает историю в сокращенном виде:
  - `$ git log --pretty=format:"%h - %an, %ar : %s"`
- Список коммитов, сделанных за последние две недели:
  - `$ git log --since=2.weeks`

# Операции отмены

- Если вы хотите переделать коммит — внесите необходимые изменения, добавьте их в индекс и сделайте коммит ещё раз, указав параметр `--amend`:
  - `$ git commit --amend`
- Например, если вы сделали коммит и поняли, что забыли проиндексировать изменения в файле, который хотели добавить в коммит, то можно сделать следующее:
  - `$ git commit -m 'initial commit'`
  - `$ git add forgotten_file`
  - `$ git commit --amend`

В итоге получится единый коммит — второй коммит заменит результаты первого.

# Отмена изменений в файле

- Что делать, если вы поняли, что не хотите сохранять свои изменения файла CONTRIBUTING.md? Как можно просто отменить изменения в нём (**git status** подсказывает):

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: CONTRIBUTING.md

- Как отменить существующие изменения?
  1. \$ git checkout -- CONTRIBUTING.md
  2. \$ git restore CONTRIBUTING.md

# Отмена индексации файла

- Например, вы изменили два файла и хотите добавить их в разные коммиты, но случайно выполнили команду `git add *` и добавили в индекс оба (**git status** тоже подсказывает):

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

renamed: README.md -> README

modified: CONTRIBUTING.md

- Как исключить из индекса один из них?
  1. `$ git reset HEAD CONTRIBUTING.md`
  2. `$ git restore --staged CONTRIBUTING.md`

# Отмена коммита (возврат)

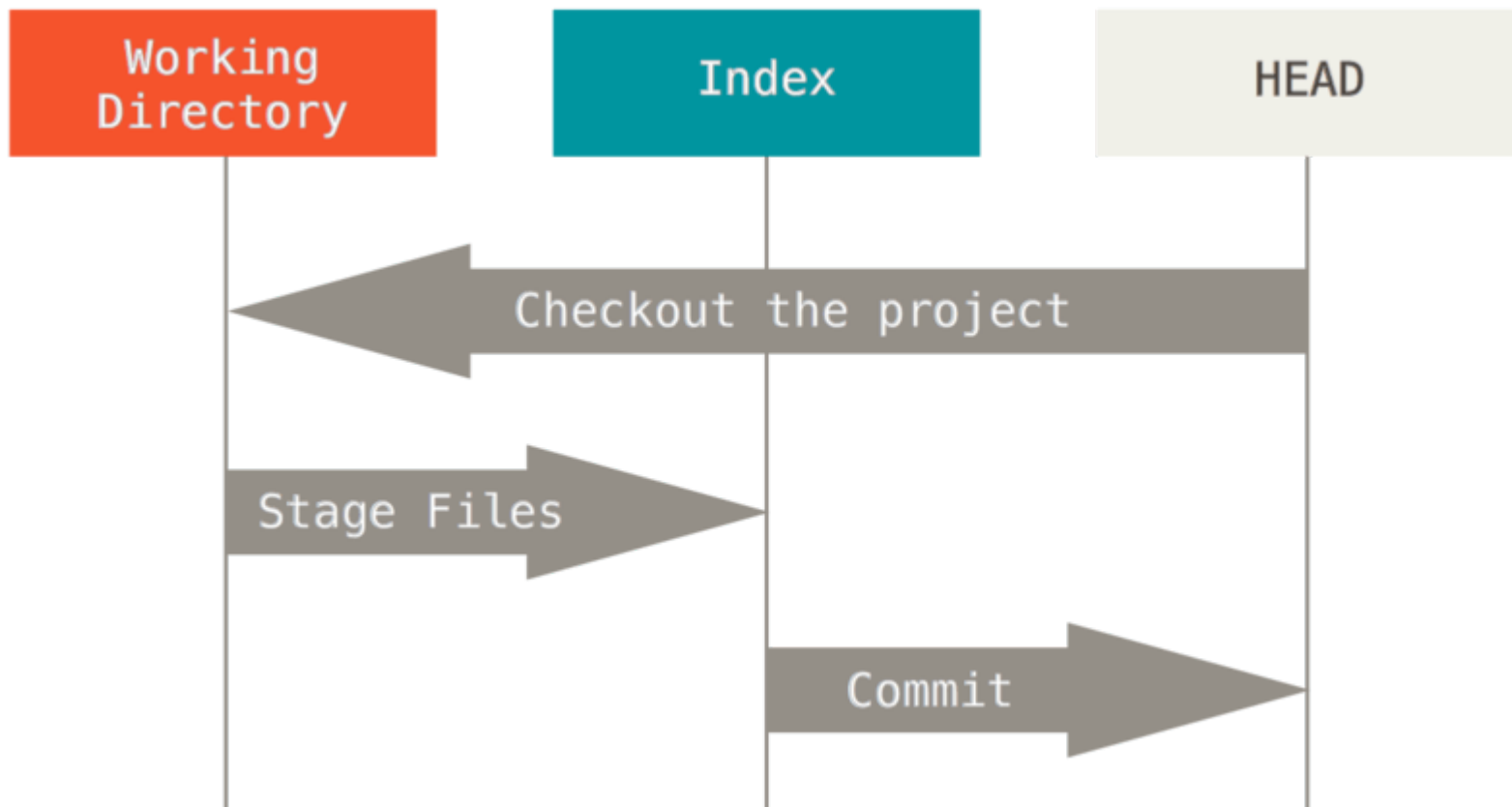
- Отмена последнего коммита (возврат на 1 коммит):
  - `$ git reset --soft HEAD^1`
- Отмена последнего коммита и изменения файла:
  - `$ git reset --hard HEAD^1`

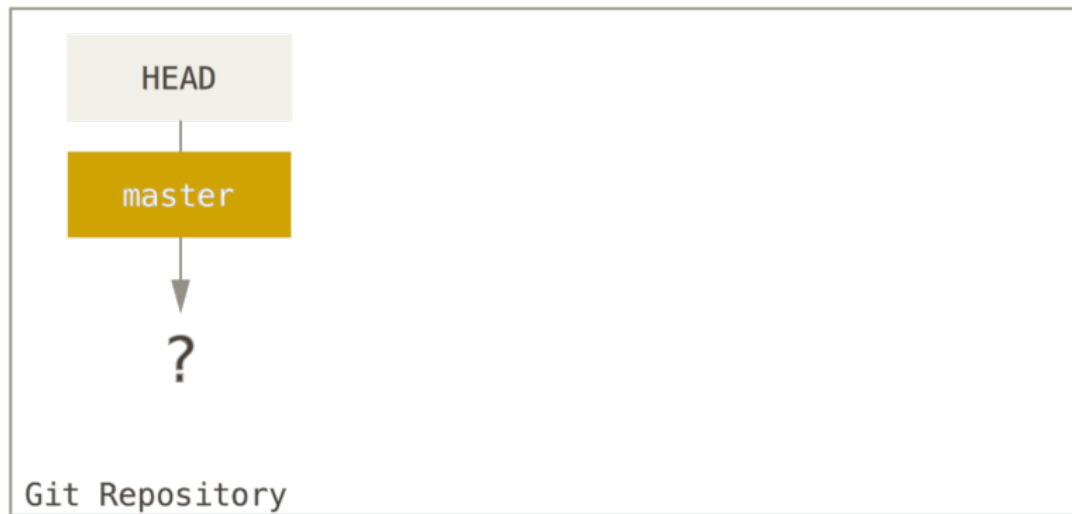
# Раскрытие тайн reset

- Давайте поговорим о **reset** и **checkout**. Эти команды кажутся самыми непонятными из всех, которые есть в Git, когда вы в первый раз сталкиваетесь с ними. Они делают так много, что попытки по-настоящему их понять и правильно использовать кажутся безнадёжными. Для того, чтобы всё же достичь этого, мы советуем воспользоваться простой аналогией.
- Разобраться с командами **reset** и **checkout** будет проще, если считать, что Git управляет содержимым трёх различных деревьев. Здесь под “деревом” мы понимаем “набор файлов”, а не специальную структуру данных. (В некоторых случаях индекс ведет себя не совсем так, как дерево, но для наших текущих целей его проще представлять именно таким.)

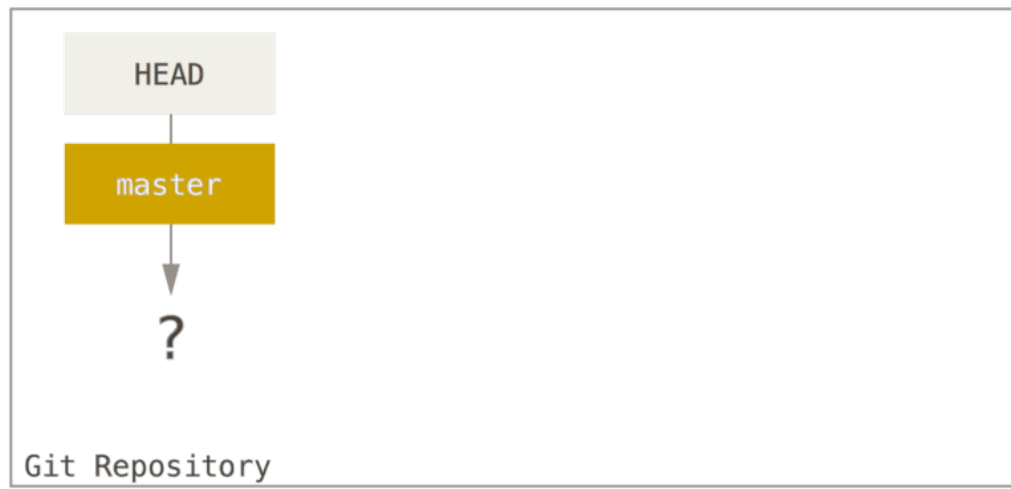


# Три дерева



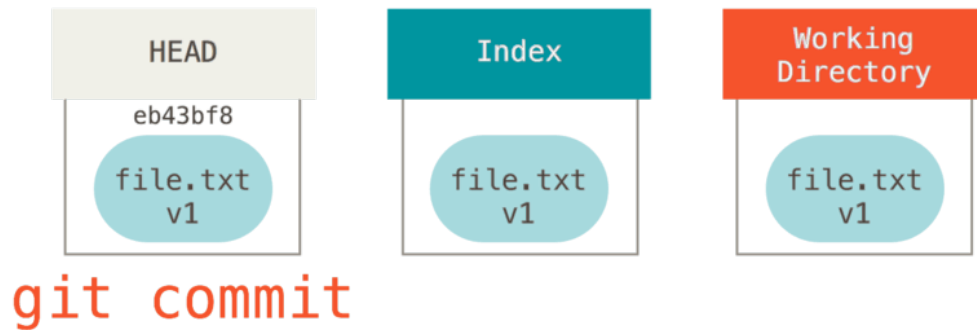
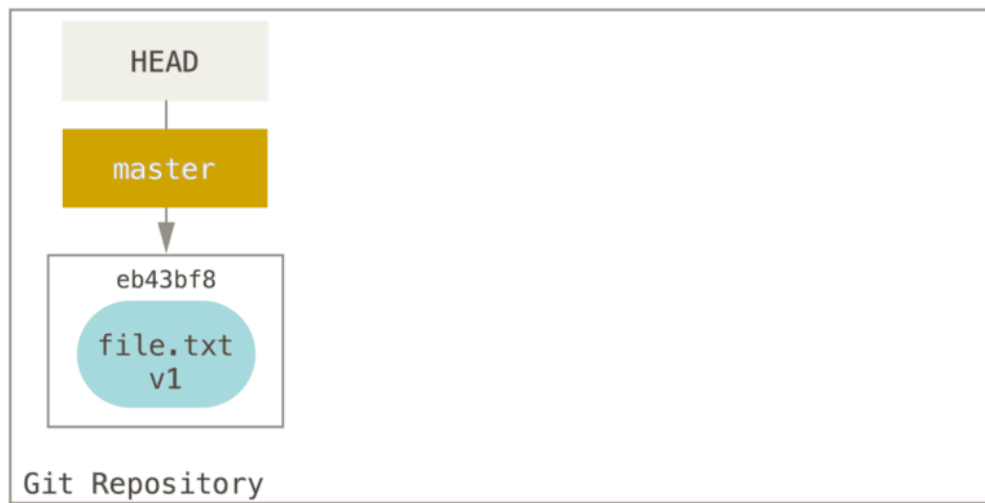


На данном этапе только дерево Рабочего Каталога содержит данные.



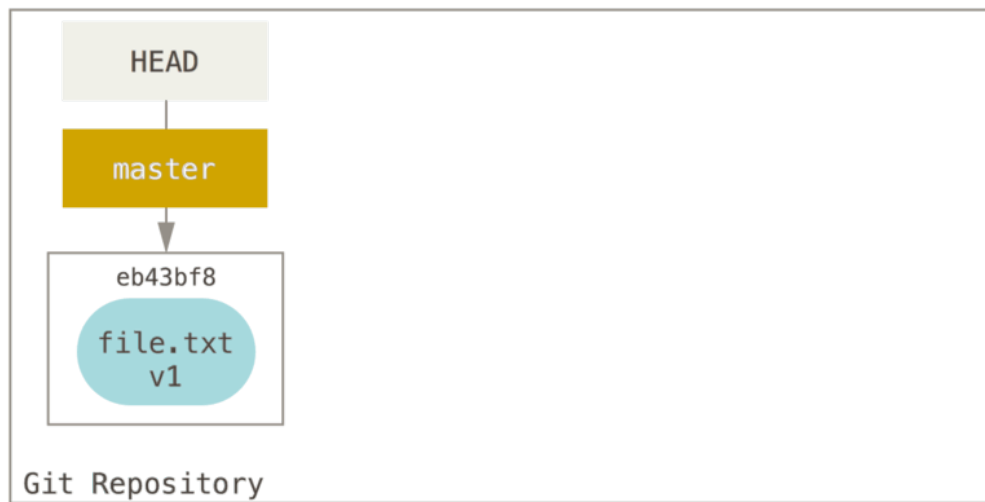
git add

Теперь мы хотим закоммитить этот файл, поэтому мы используем `git add` для копирования содержимого Рабочего Каталога в Индекс



Затем, мы выполняем команду `git commit`, которая сохраняет содержимое Индекса как неизменяемый снимок, создает объект коммита, который указывает на этот снимок, и обновляет `master` так, чтобы он тоже указывал на этот коммит.

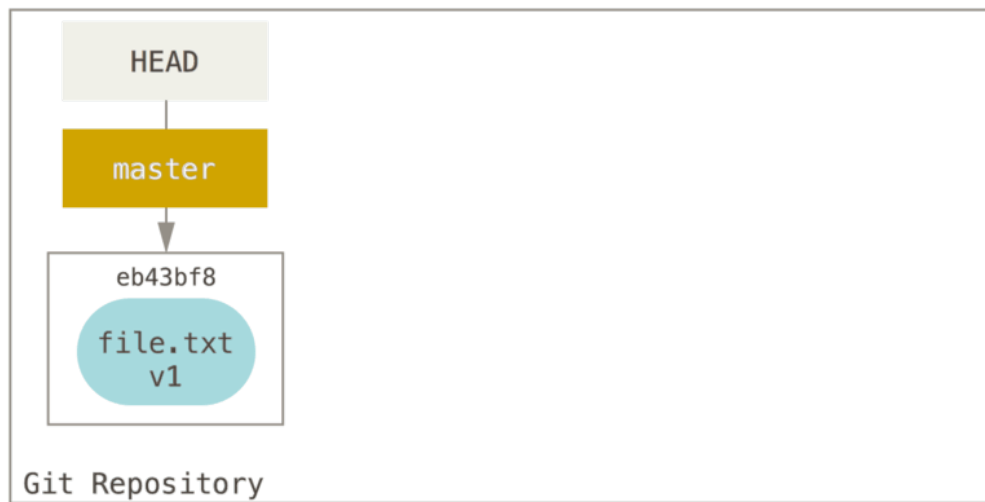
Если сейчас выполнить `git status`, то мы не увидим никаких изменений, так как все три дерева одинаковые.



edit file

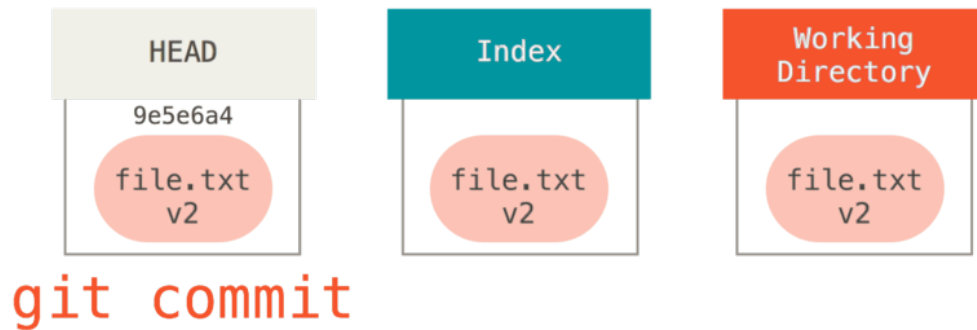
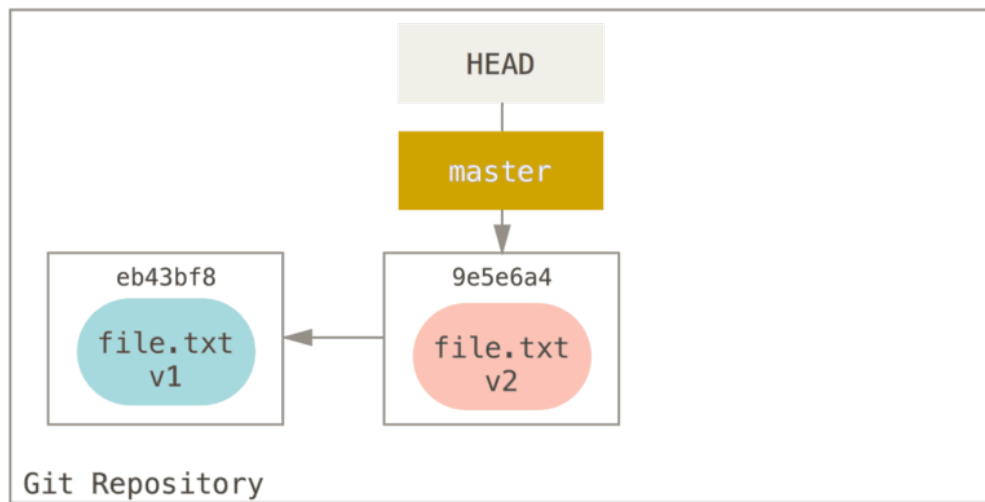
Теперь мы хотим внести изменения в файл и закоммитить его. Мы пройдем через всё ту же процедуру; сначала мы отредактируем файл в нашем рабочем каталоге.

Если сейчас мы выполним `git status`, то увидим, что файл выделен красным в разделе “Изменения, не подготовленные к коммиту”.



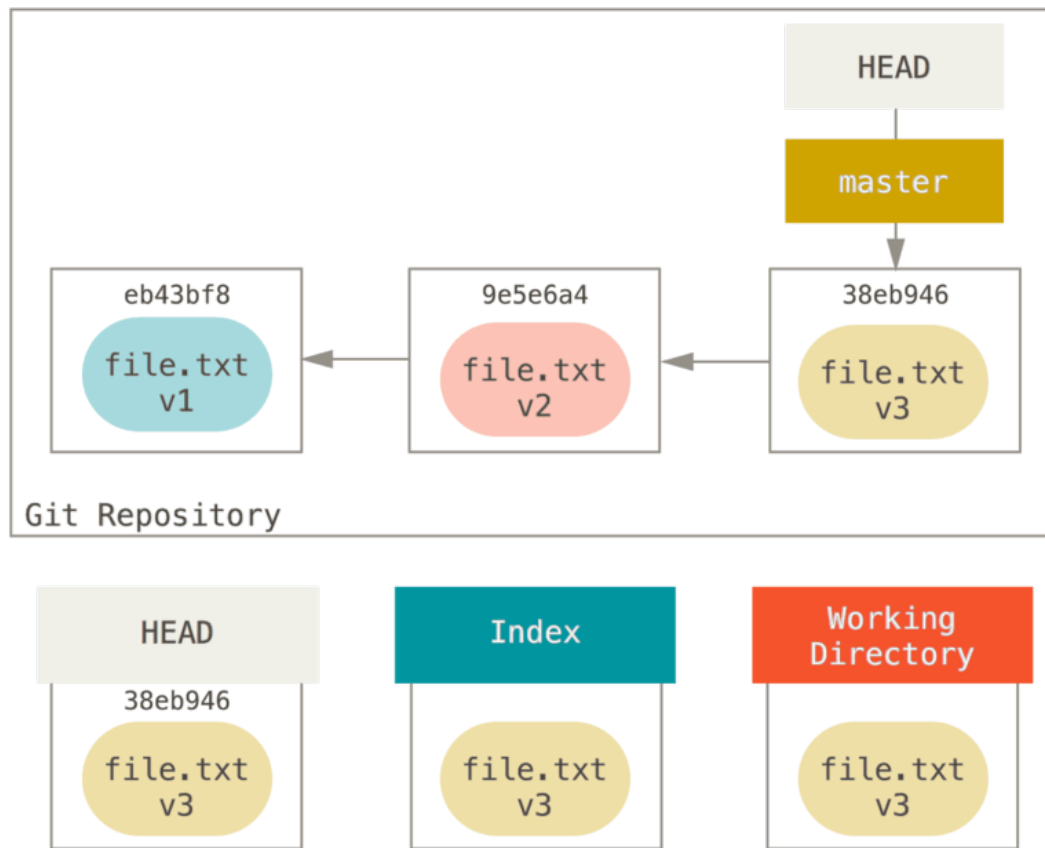
`git add`

Если сейчас мы выполним `git status`, то увидим, что этот файл выделен зелёным цветом в разделе “Изменения, которые будут закоммичены”, так как Индекс и HEAD различны



Сейчас команда `git status` не показывает ничего, так как снова все три дерева одинаковые.

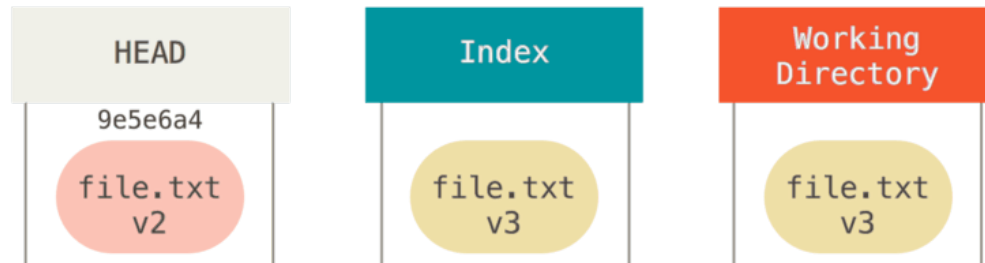
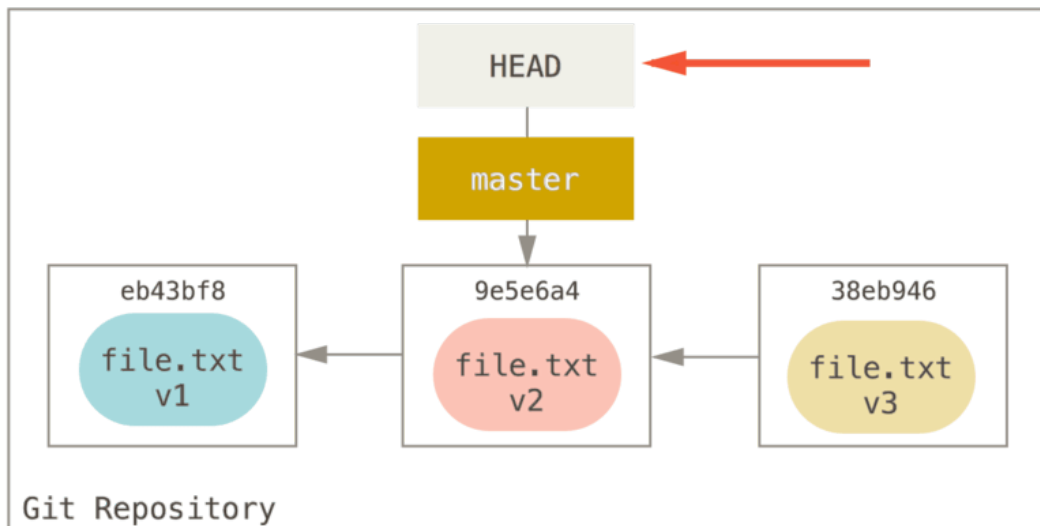
Команда **reset** становится более понятной, если рассмотреть её с учётом вышеизложенного.



Назначение **reset**: Перемещение HEAD, обновление индекса, обновление рабочего Католога

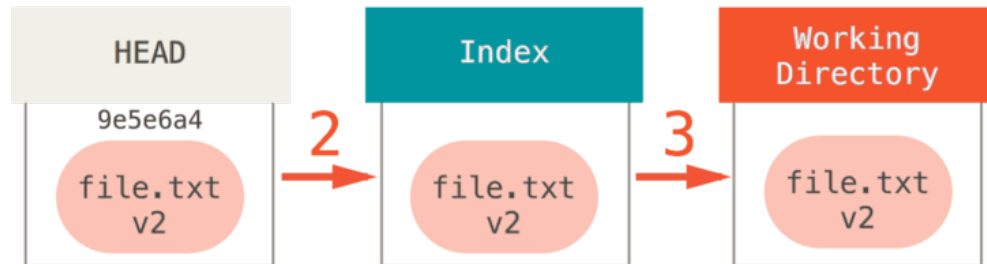
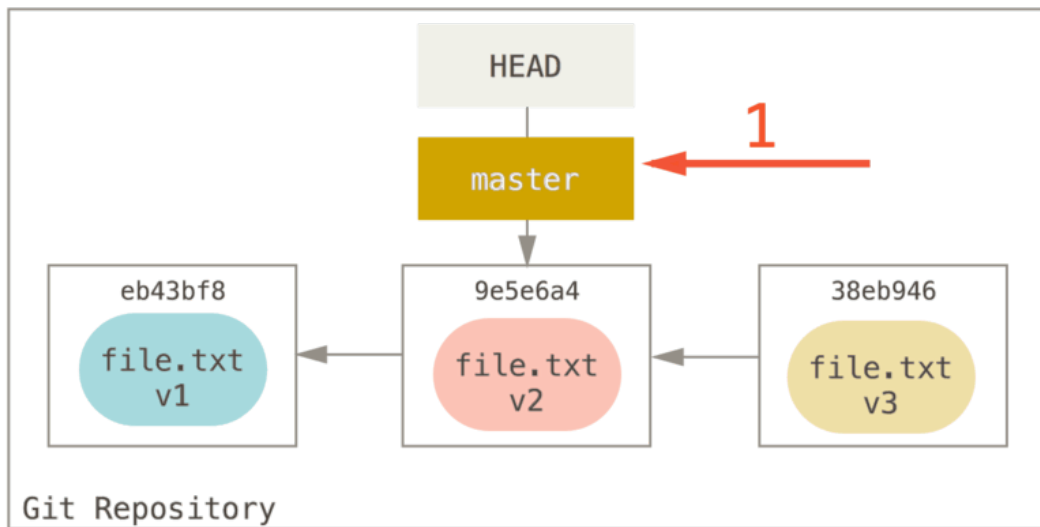


# Перемещение HEAD (--soft)



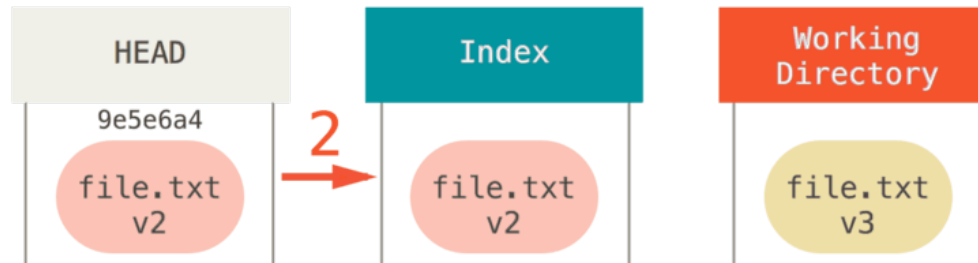
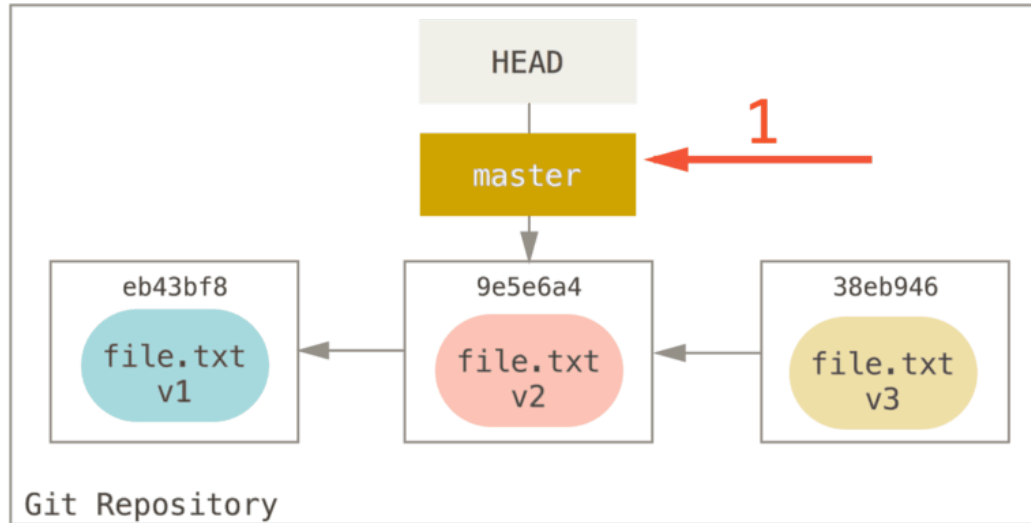
`git reset --soft HEAD~`

# Обновление Индекса (--mixed)



`git reset --hard HEAD~`

# Обновление Рабочего каталога (--hard)



`git reset [--mixed] HEAD~`

# Основные операции отмены

- Изменили файл **test.cpp**. Отменить изменения в файле:
  1. `$ git checkout -- test.cpp`
  2. `$ git restore test.cpp`
- Добавили изменения в **stage (add)**. Убрать из **stage::**:
  1. `$ git reset HEAD test.cpp`
  2. `$ git restore --staged test.cpp`
- Сделали коммит в локальный репозиторий. Отменить коммит:
  - `$ git reset --soft HEAD^1`
- Отменить коммит и изменения файлов:
  - `$ git reset --hard HEAD^1`

# Псевдонимы в Git

- Вот несколько примеров псевдонимов, которые вы, возможно, захотите задать:
  1. `$ git config --global alias.st status`
  2. `$ git config --global alias.co checkout`
  3. `$ git config --global alias.ci commit`
  4. `$ git config --global alias.br branch`
  5. `$ git config --global alias.unstage 'reset HEAD -'`
  6. `$ git config --global alias.ll 'log --pretty=format:"%h - %an, %ar : %s"'`