Дорожная карта

0. Приступаем
1. Введение в Git
2. Начало работы с Git
3. Просмотр истории
4. Ветвление
5. Слияние
6. Отмена изменений 👈
7. Рабочий процесс

Дорожная карта (продолжение)

8. Работа в команде
9. Метки
10. Последние штрихи
11. Завершаем

Отмена изменений

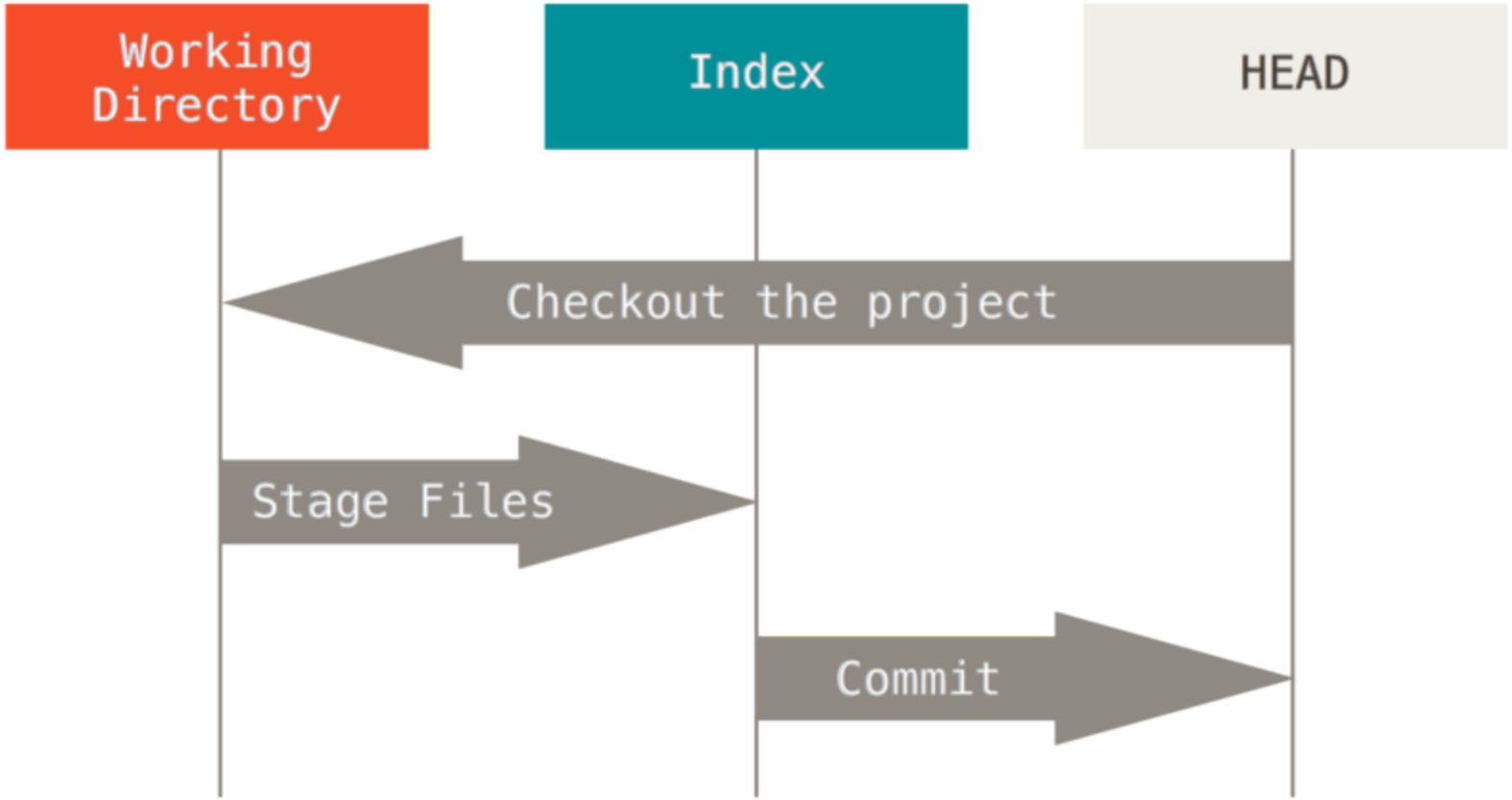
В Git нет команды отмены `undo`, которая обычно встречается в программах. Однако есть действия, противоположные другим. За счёт этого и достигается эффект отмены.

Не все действия отмены можно отменить!

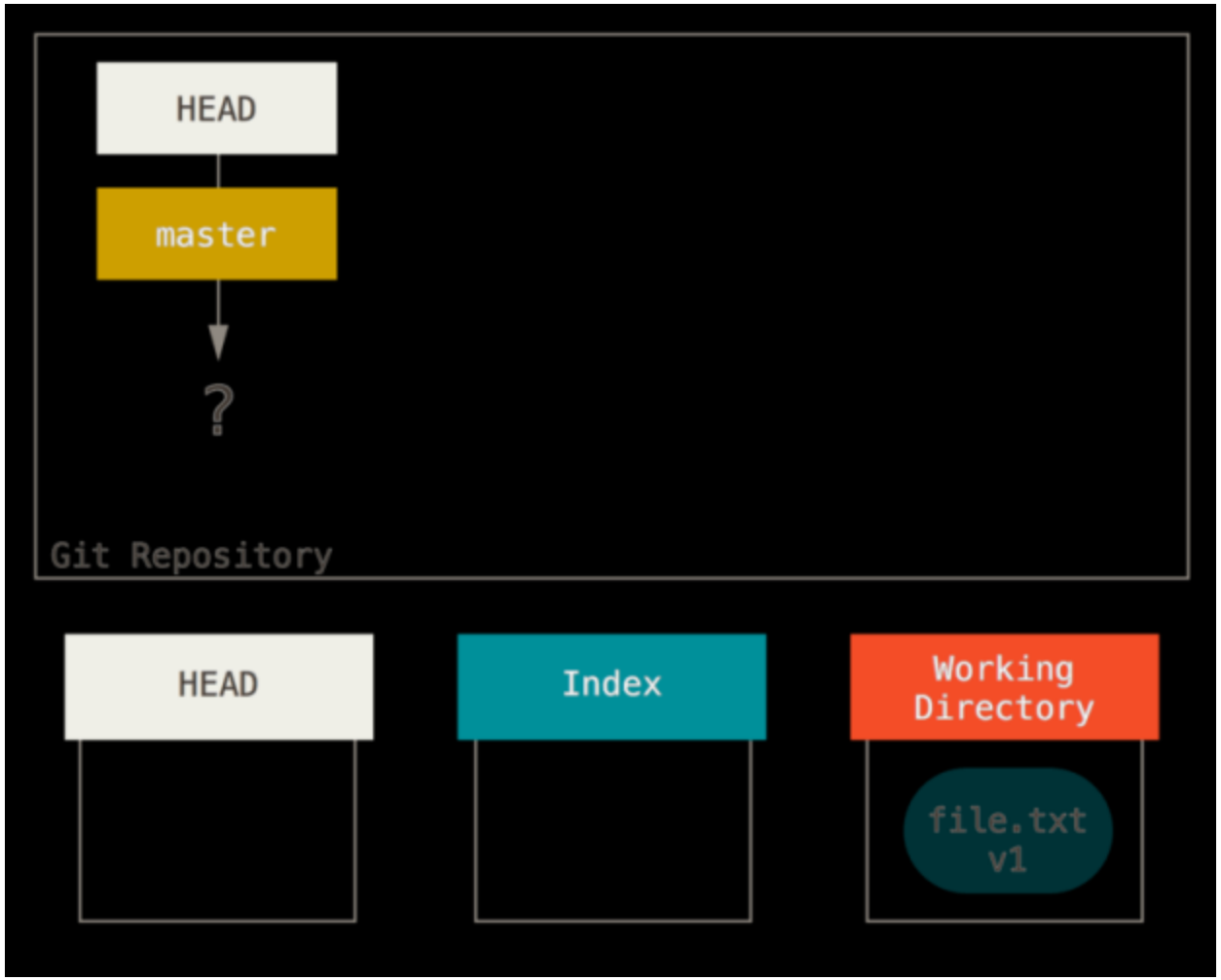
Три дерева

Дерево	Назначение
HEAD	Снимок последнего коммита, родитель следующего
Индекс	Снимок следующего намеченного коммита
Рабочий Каталог	Песочница

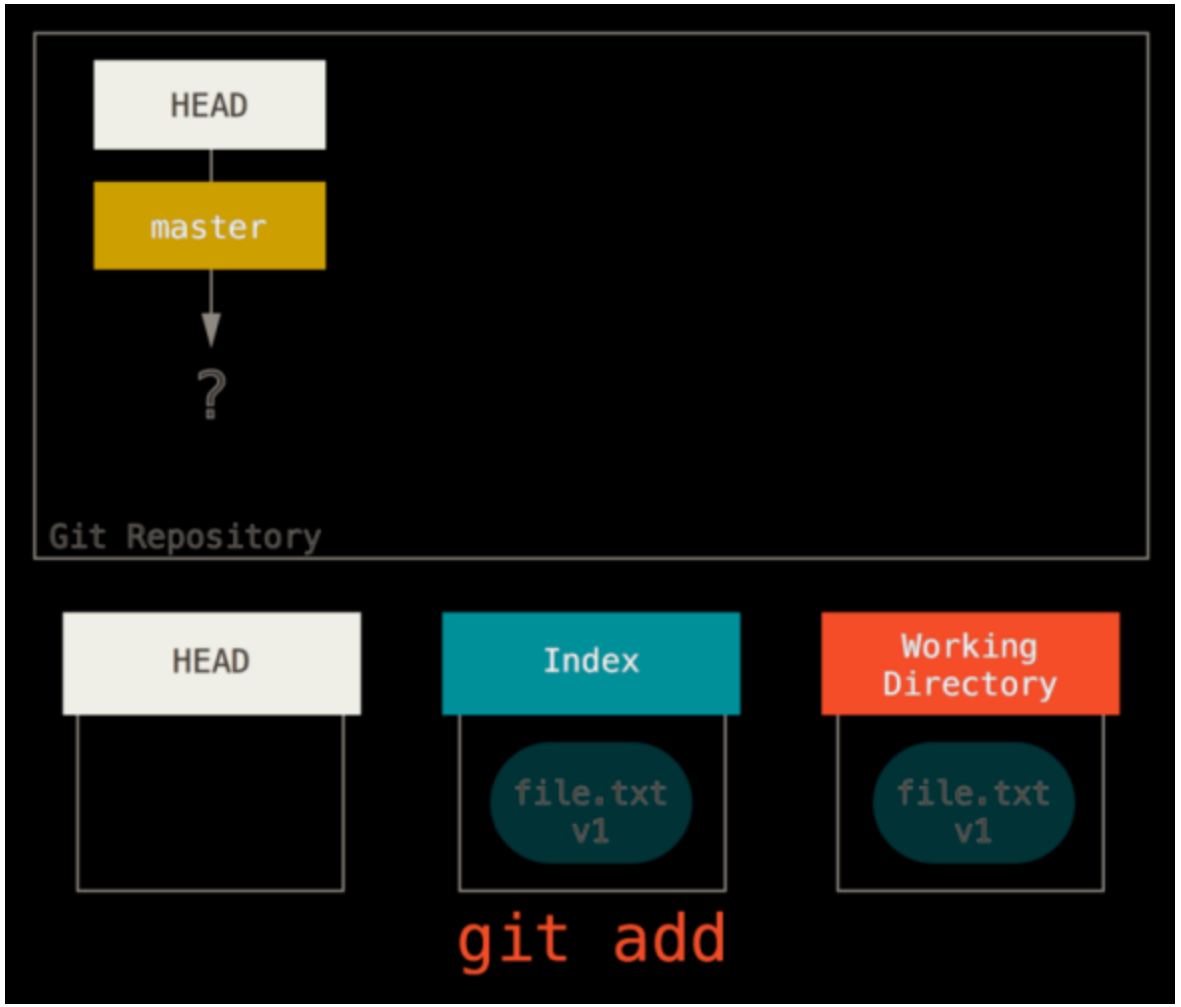
Технологический процесс



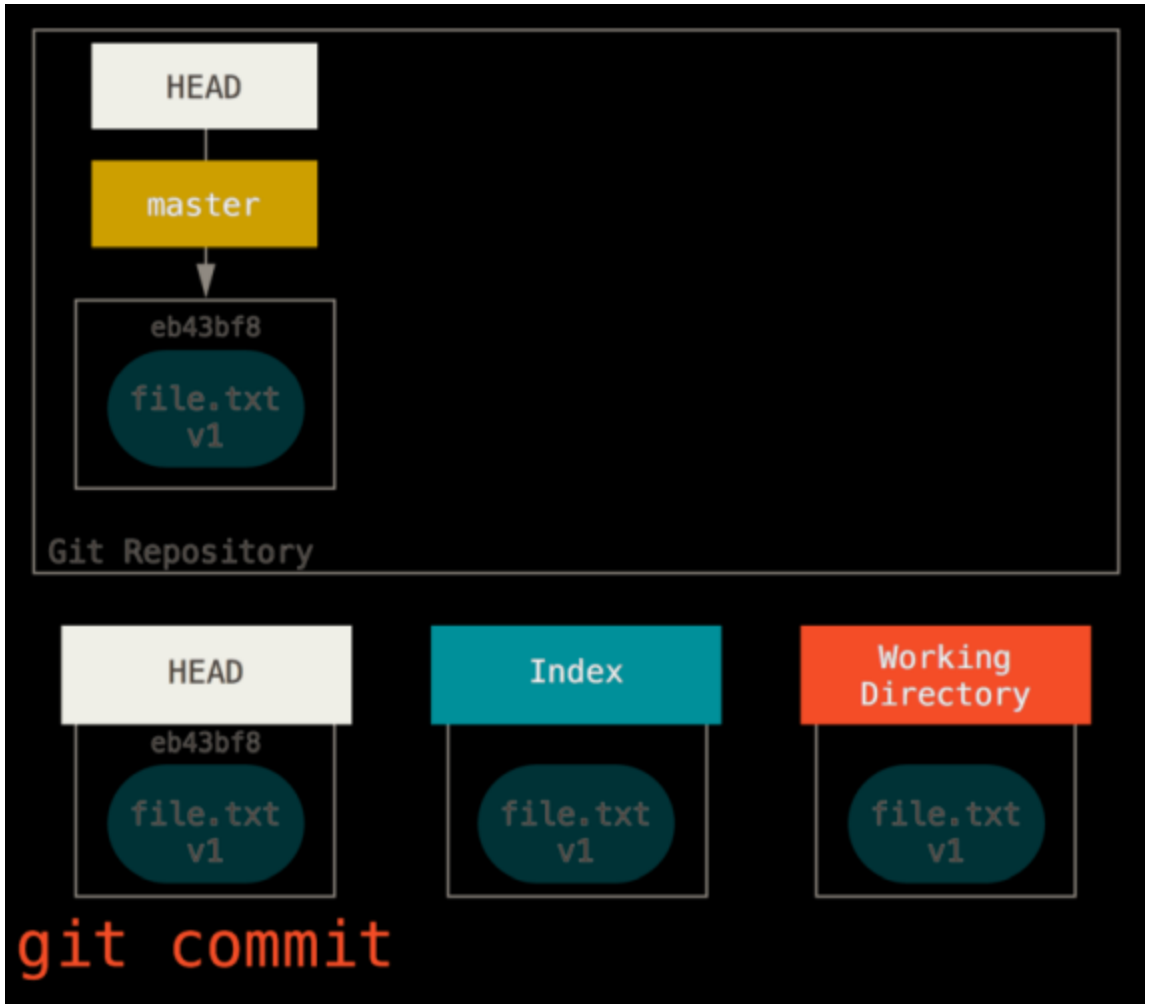
Технологический процесс



Технологический процесс

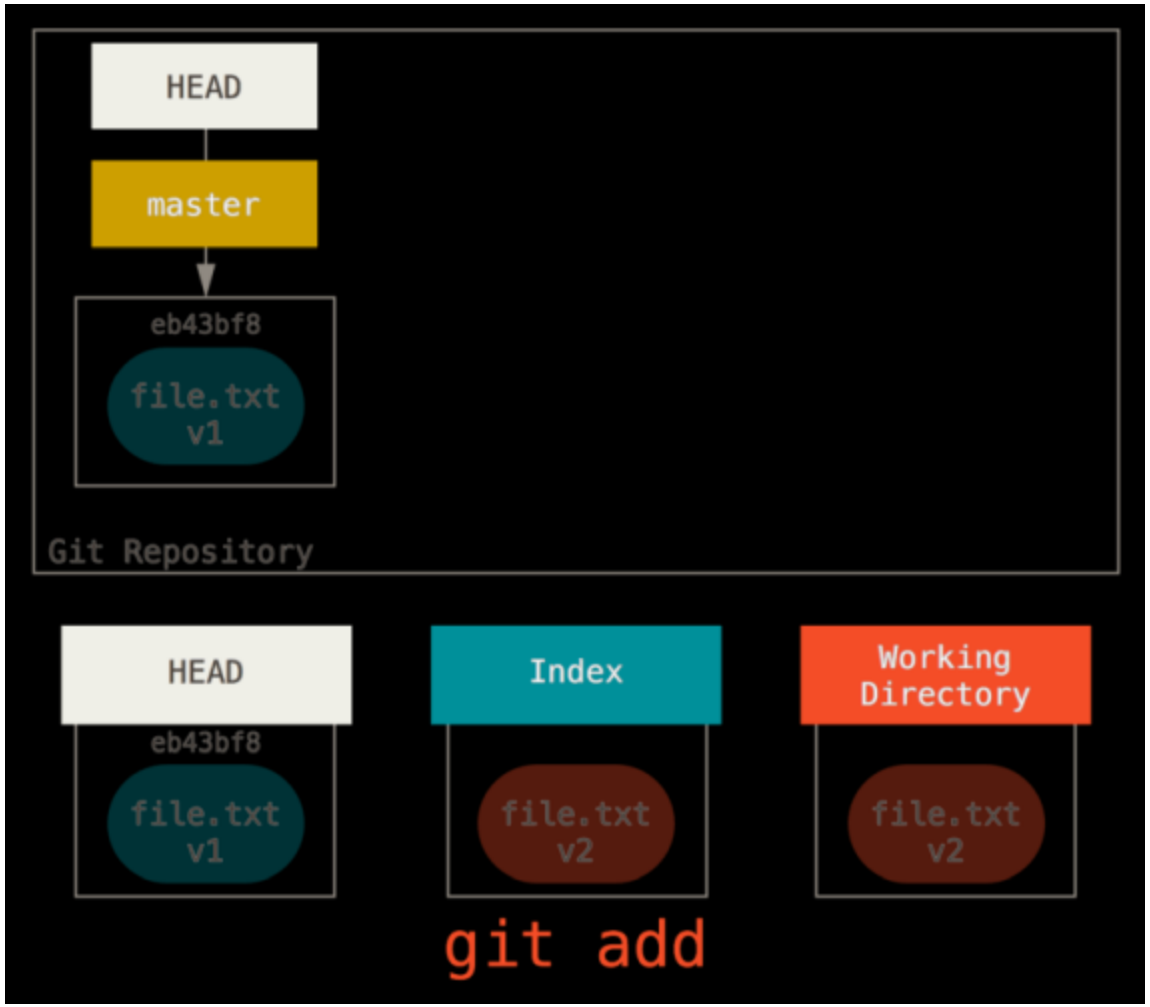


Технологический процесс

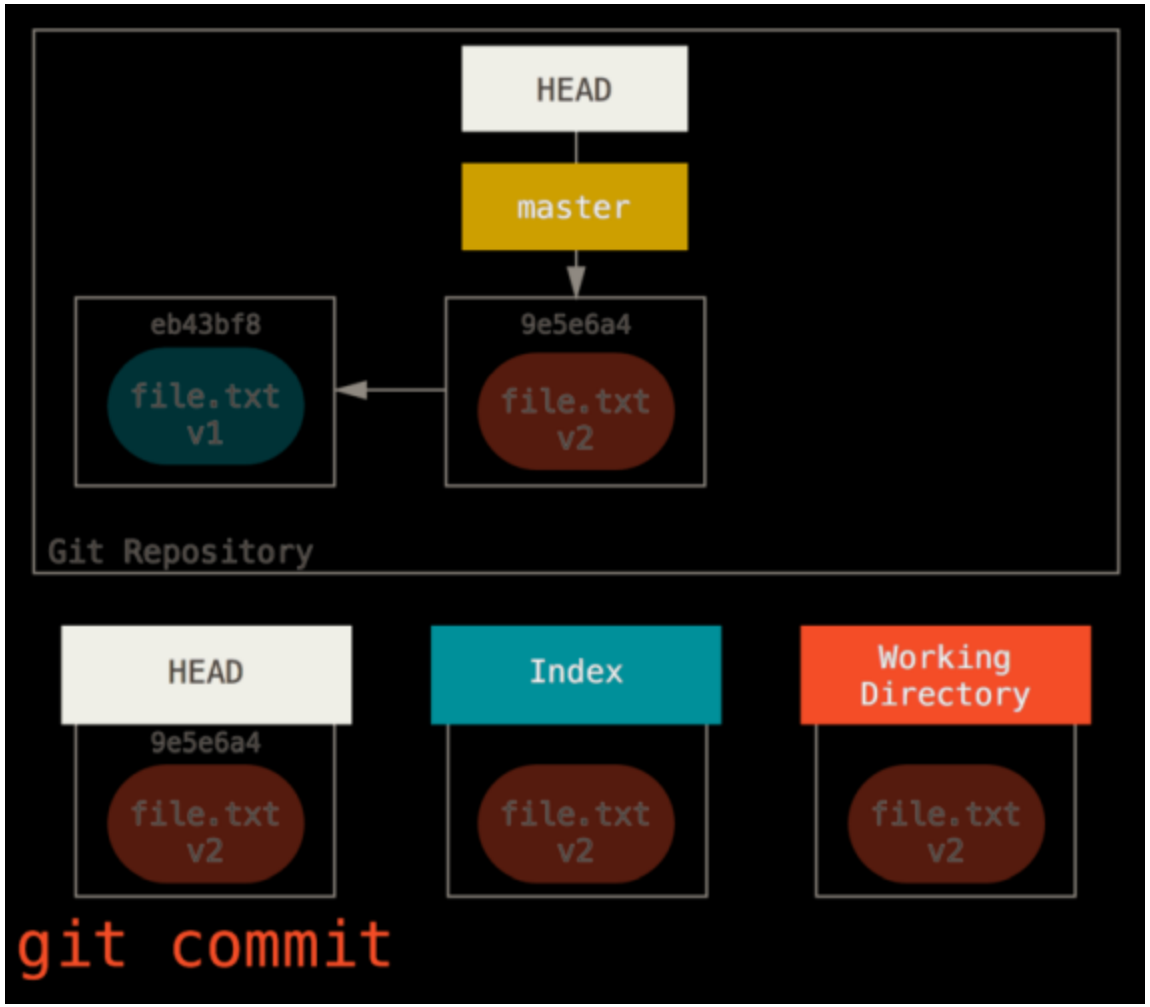


Технологический процесс

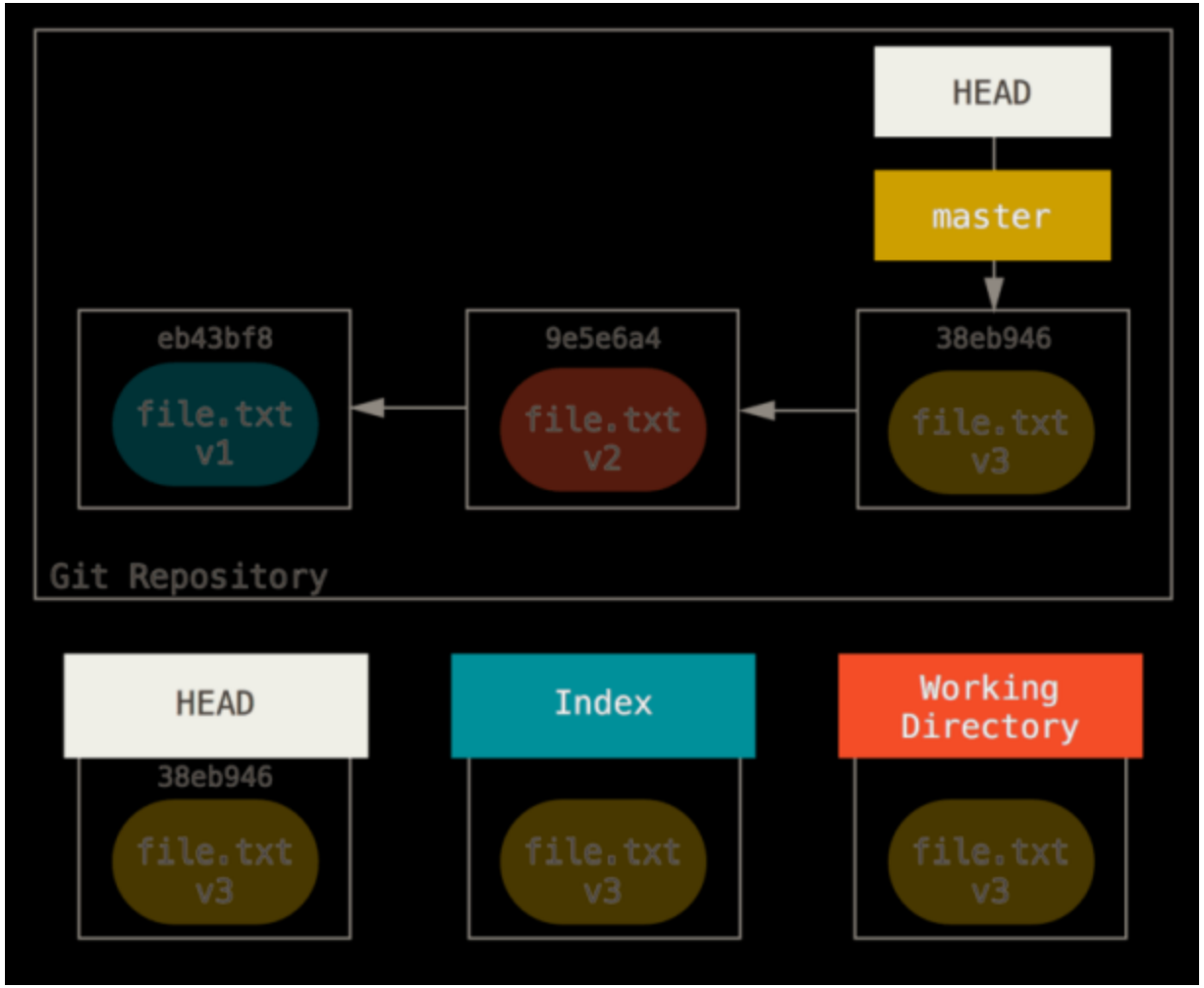
Технологический процесс



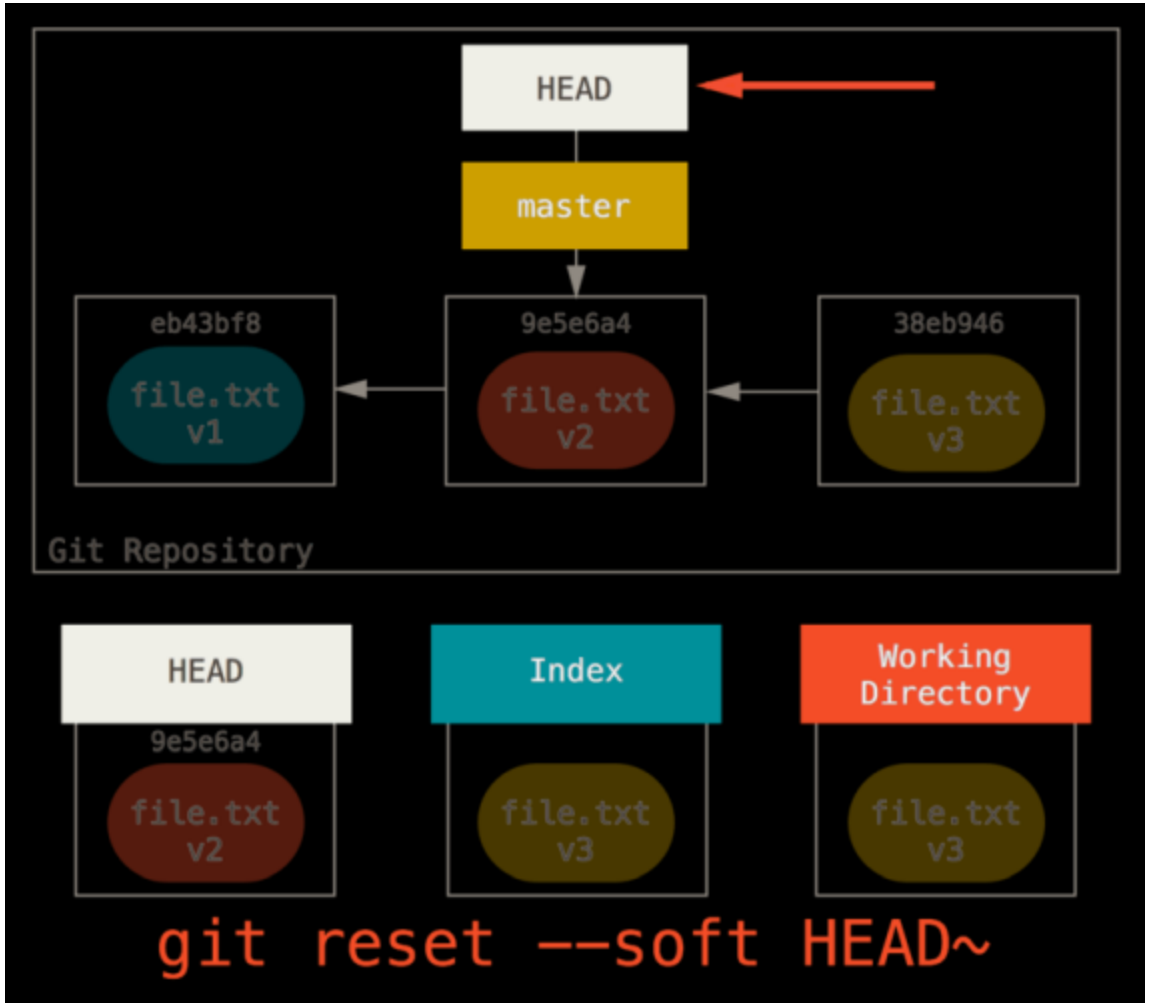
Технологический процесс



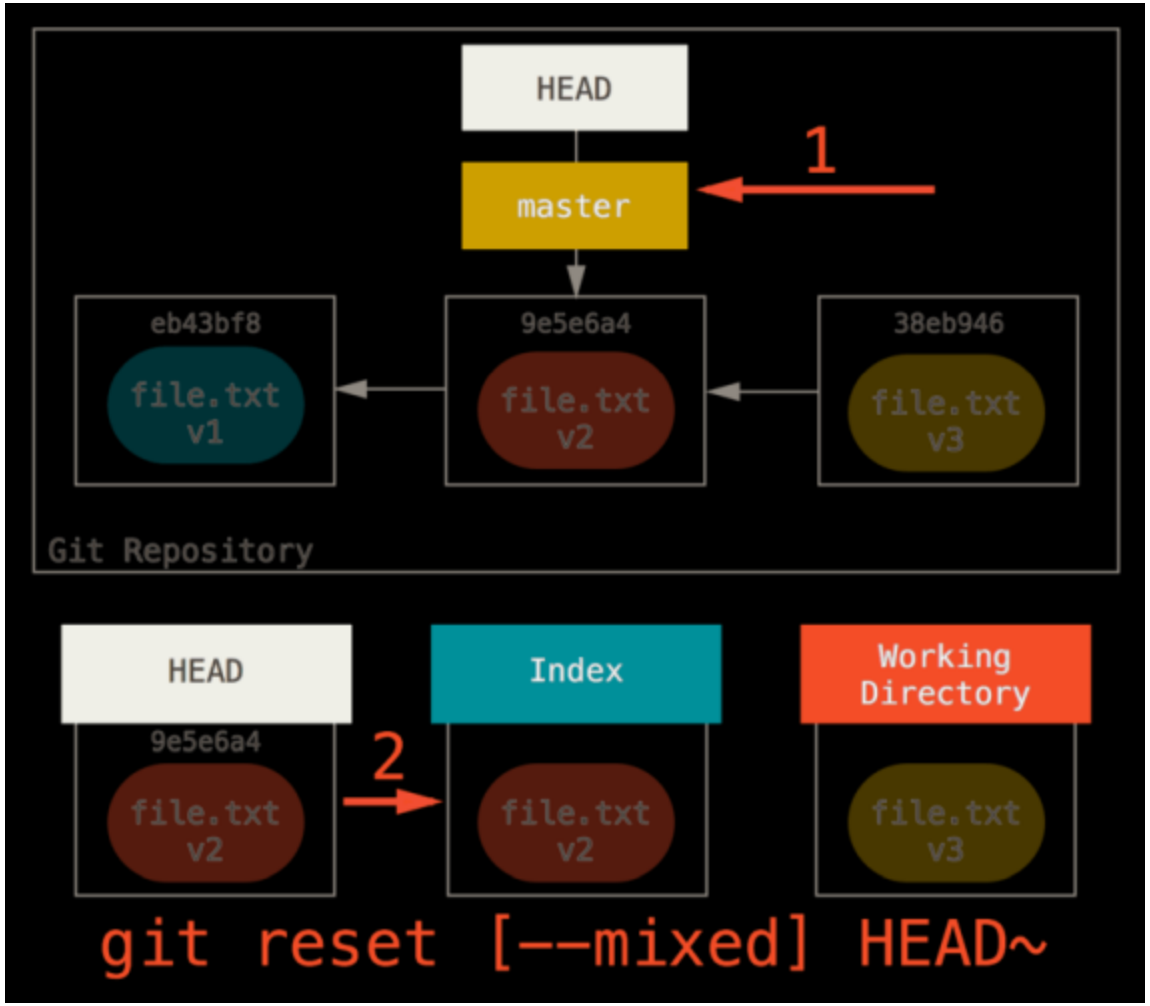
`git reset` на уровне коммита



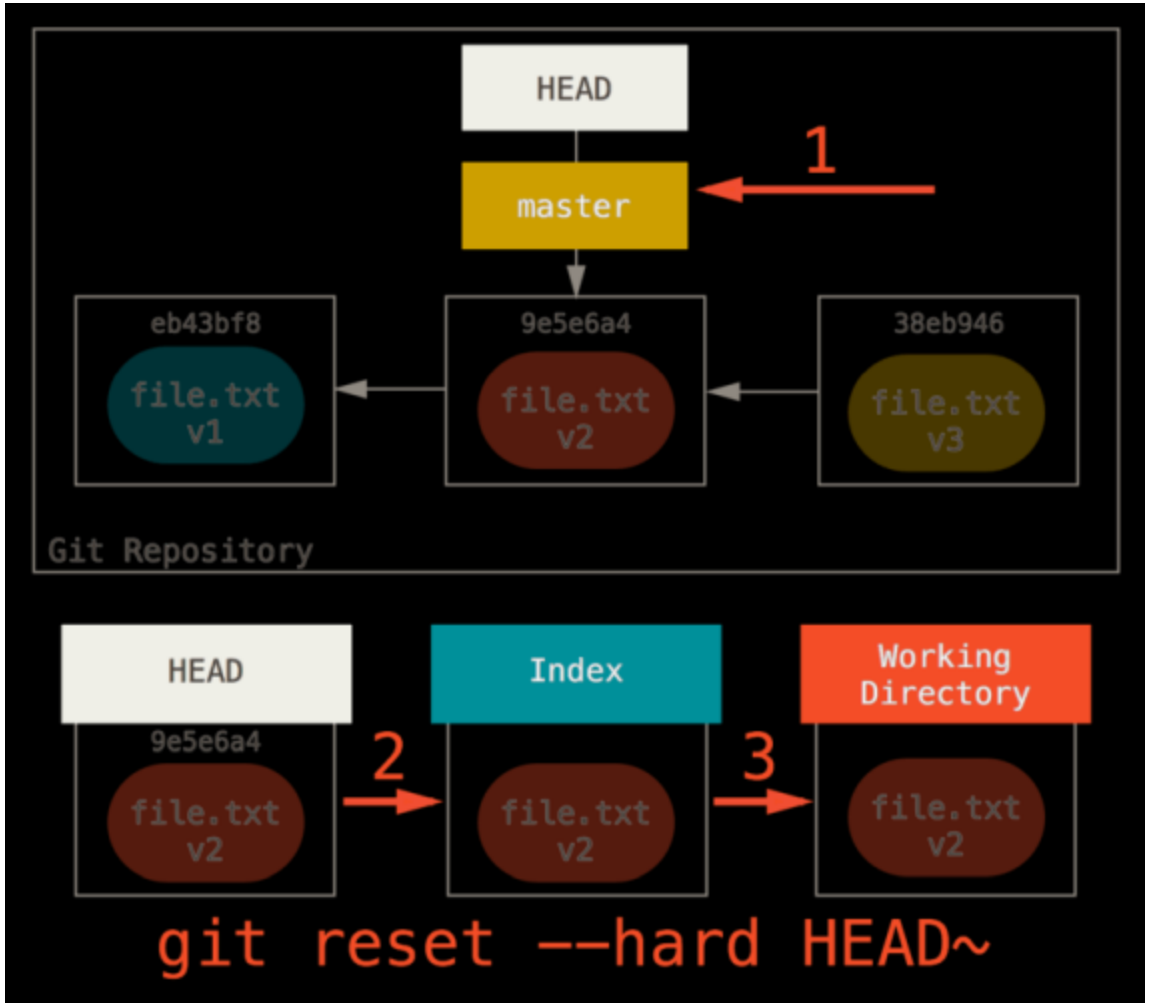
Шаг 1: Перемещение `HEAD` (`--soft`)



Шаг 2: Обновление Индекса (`--mixed`)



Шаг 3: Обновление Рабочего Каталога (`--hard`)

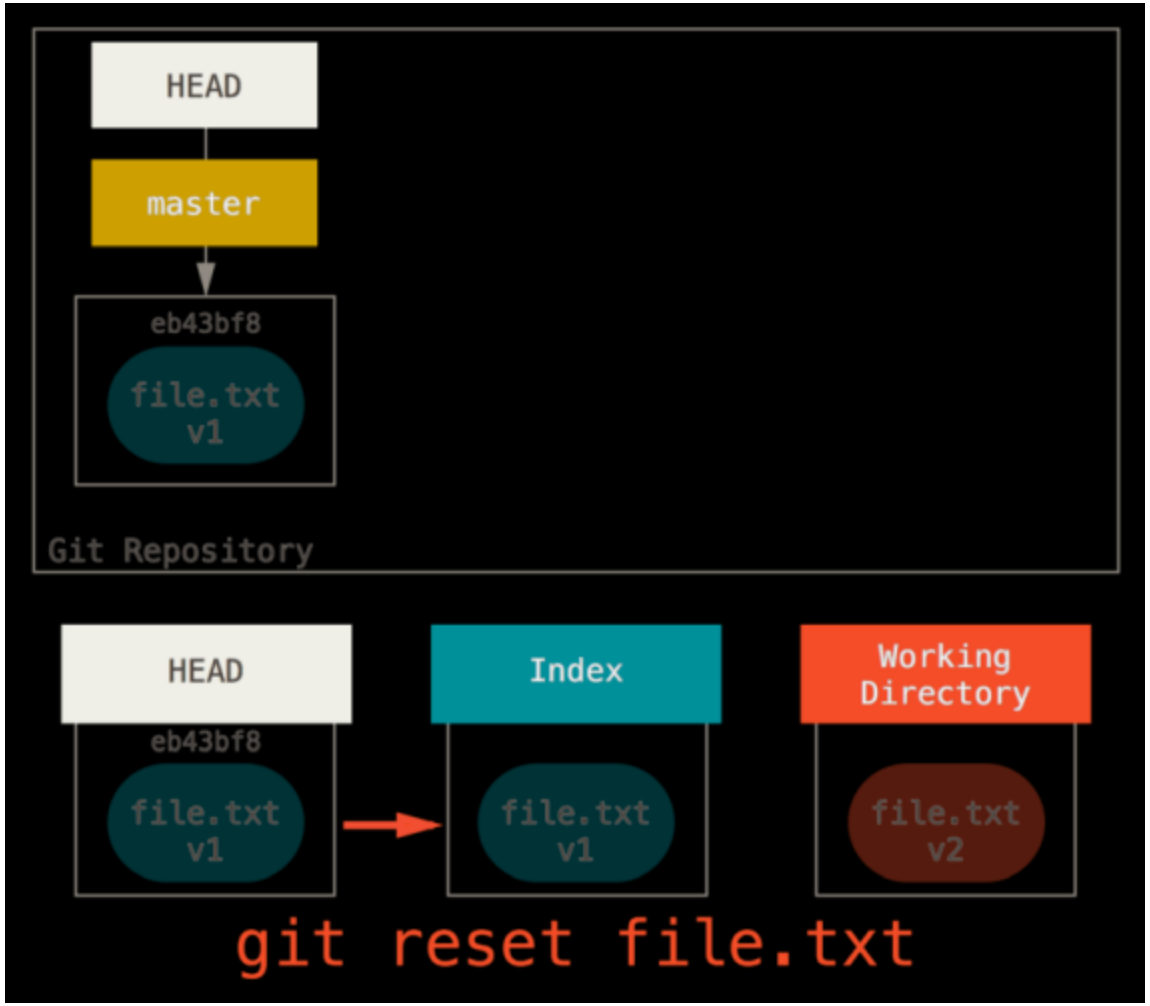


Резюме по команде `reset` на уровне коммита

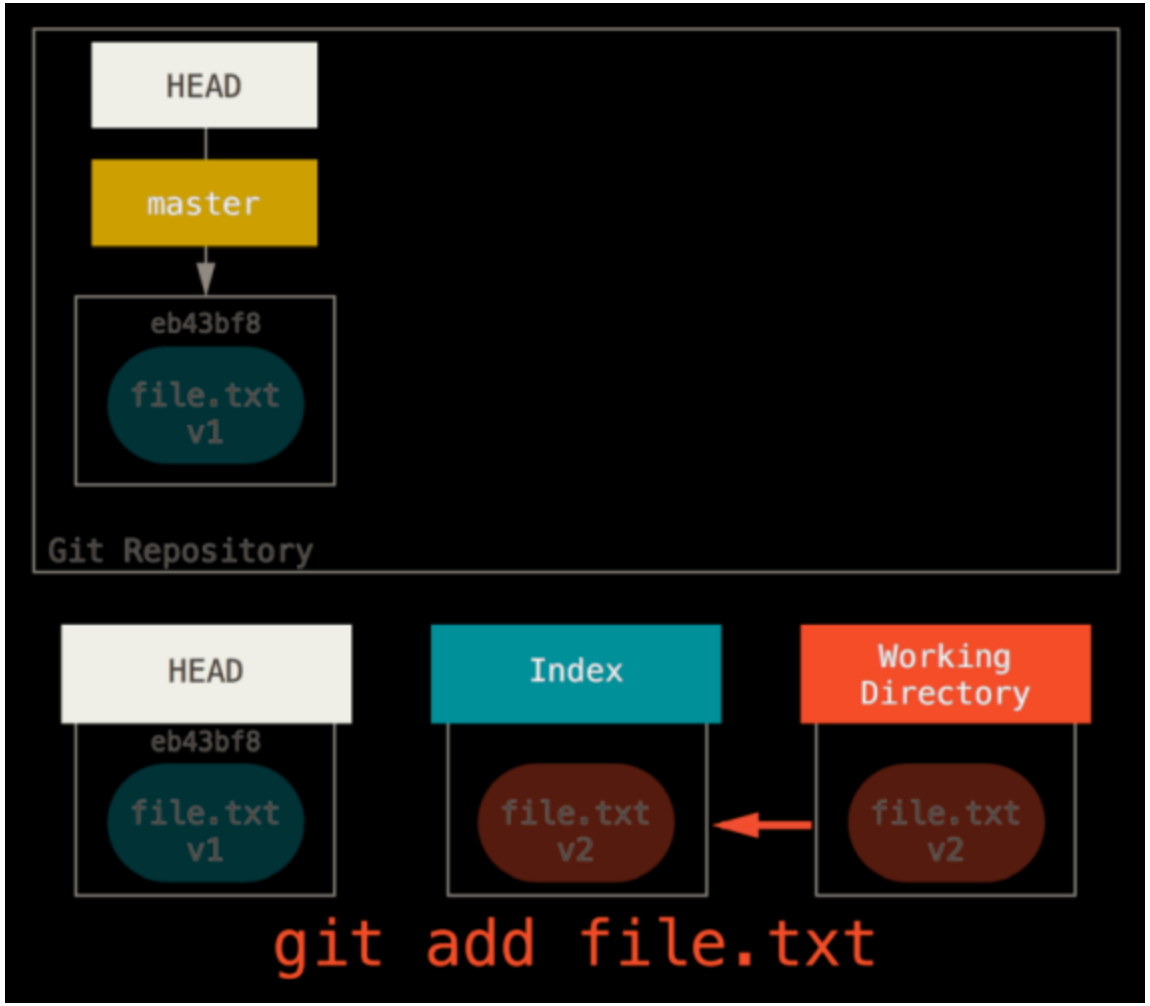
Команда `git reset` в заранее определенном порядке перезаписывает три дерева Git, останавливаясь тогда, когда вы ей скажете:

1. Перемещает ветку, на которую указывает `HEAD` (останавливается на этом, если указана опция `--soft`)
2. Делает Индекс таким же как и `HEAD` (останавливается на этом, если не указана опция `--hard`)
3. Делает Рабочий Каталог таким же как и Индекс.

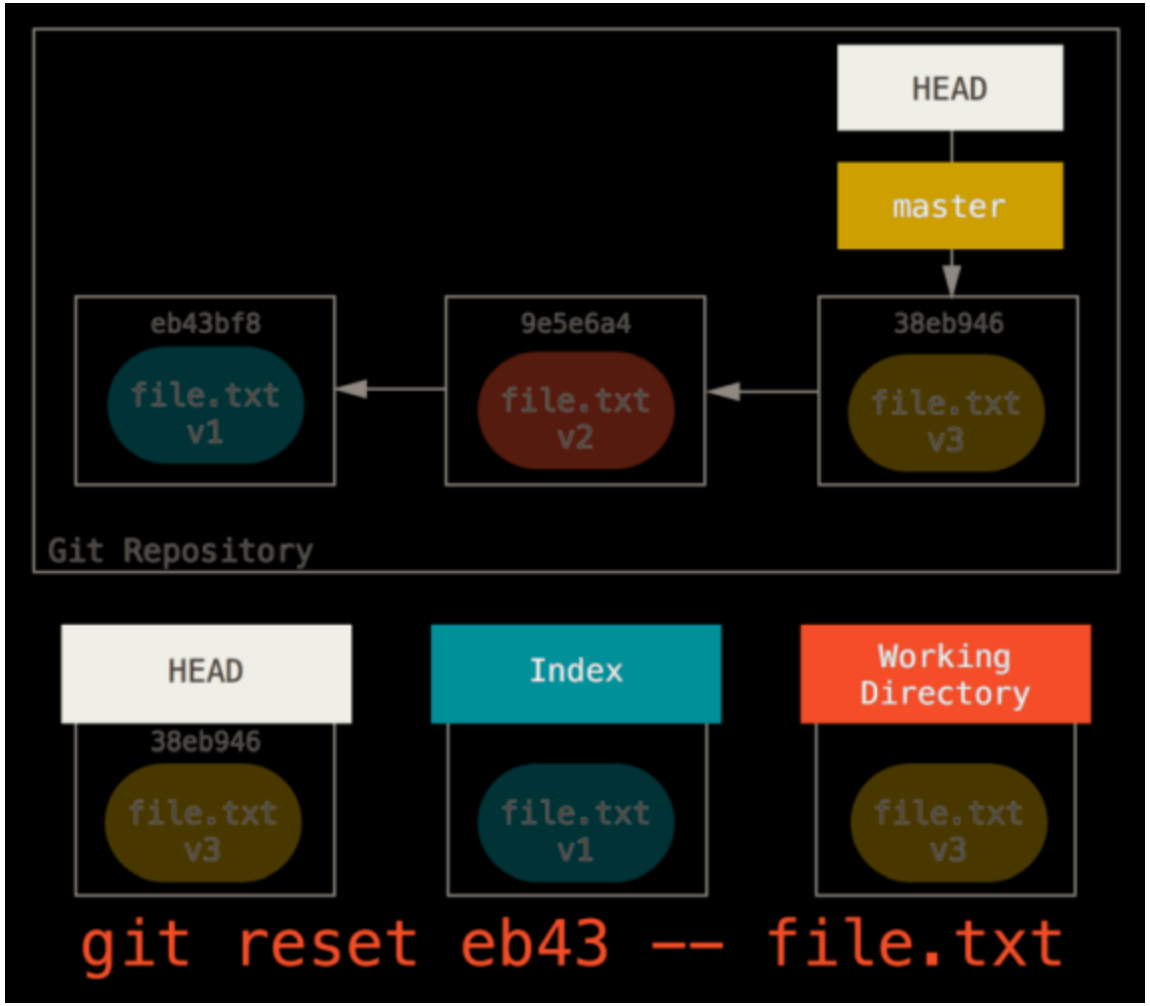
`git reset` на уровне файла



Противоположность `git add`



`git reset` на уровне файла (не из `HEAD`)



Практика 💪

🎯 Освоить git reset

Практика (продолжение) → `git reset --hard`

1. сделать изменения в рабочем каталоге (создаем и изменяем файлы)
2. `git reset --hard HEAD` -> изменные файлы откатываются; новые файлы остаются как есть

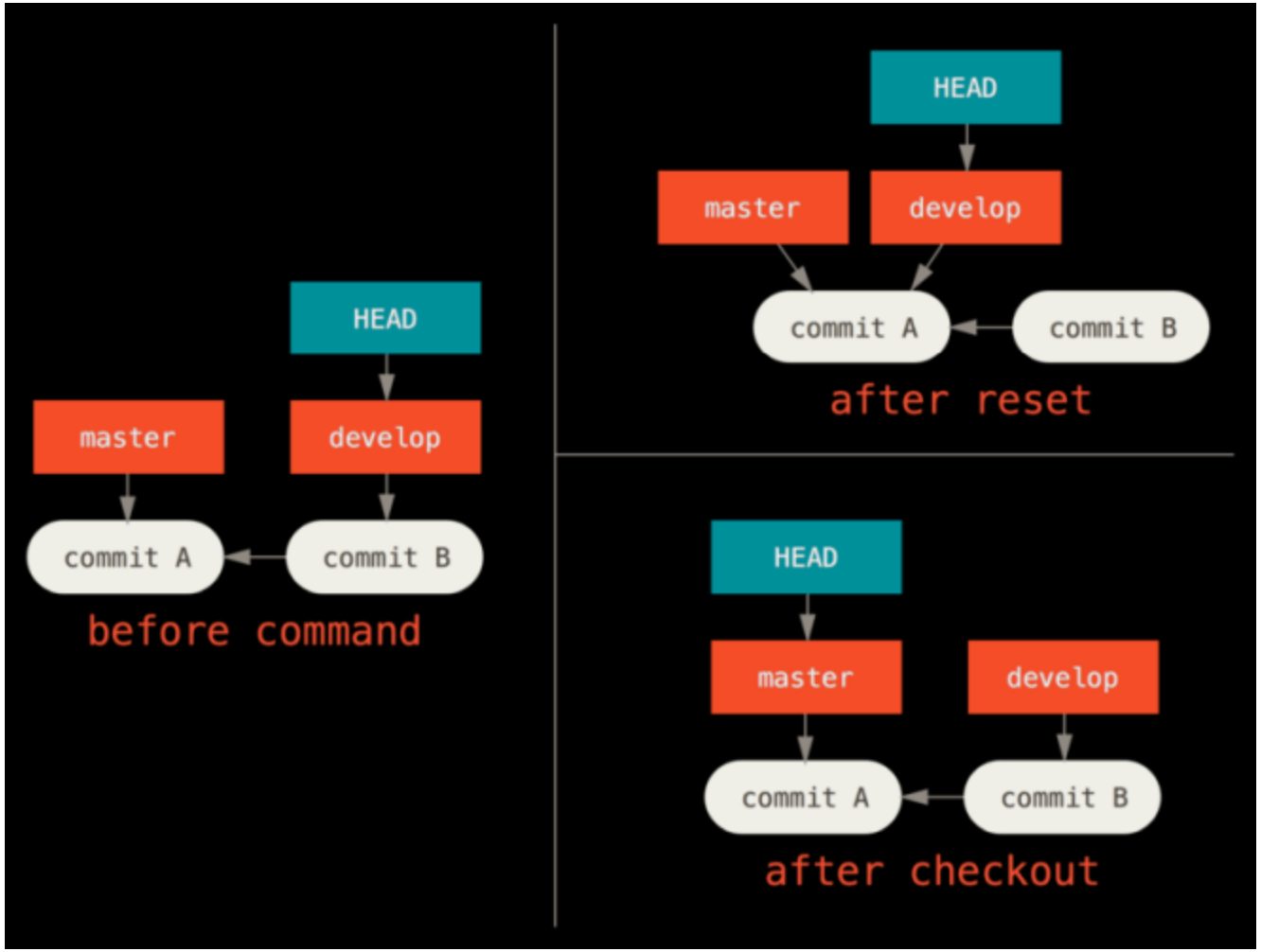
Практика (продолжение) → `git reset --hard HEAD~1`

1. Посмотреть историю `git log --oneline -4`
2. Исполняем команду `git reset --hard HEAD~1`
3. Посмотреть историю `git log --oneline -4` и увидеть, что первый коммит исчез (удалили коммит)

Практика (продолжение) → `git reset --soft HEAD~2`

1. Посмотреть историю изменений на 2 коммита `git log -2 --patch`
2. Откатиться на 2 коммита назад `git reset --soft HEAD~2`
3. Посмотреть на индекс (`git status` , `git diff --cached`) и увидеть там изменения из двух коммитов

`git checkout` на уровне коммита



`git checkout` на уровне файла

- Перемещение `HEAD` пропускается.
- Обновляет файл в индексе + в рабочем каталоге версией из коммита.
- Опция `-p` или `--patch` допускает частичное изменения файла.

Итоговое сравнение `git reset` VS. `git checkout`

	HEAD	Индекс	РК	Сохранность РК?
reset --soft [commit]	REF	NO	NO	YES
reset [commit]	REF	YES	NO	YES
reset --hard [commit]	REF	YES	YES	NO
checkout [commit]	HEAD	YES	YES	YES
reset (commit) [file]	NO	YES	NO	YES
checkout (commit) [file]	NO	YES	YES	NO

commit = На уровне коммитов = без указания путей

file = На уровне файлов = с указанием путей

git restore

Команда `git-restore` позволяет восстановить содержимое файлов:

- на диске, взяв их новое содержимое из индекса или другого коммита
- в индексе, взяв их новое содержимое из текущего или другого коммита

```
$ git restore [<options>] [--source=<tree>] [--staged] [--worktree] [--] <pathspec>…
```

Эта команда не изменяет/обновляет вашу рабочую ветку.

🧨 Команда находится считается экспериментальной и может измениться

git restore: HEAD → Рабочая область

Чтобы восстановить содержимое файла в рабочей области (копирует содержимое из `HEAD` или `--source`):

```
$ git restore README.txt
$ git restore --worktree README.txt
```

git restore: HEAD → Индекс

Чтобы восстановить содержимое файла в индексе (копирует содержимое из `HEAD` или `--source`):

```
$ git restore --staged README.txt
```

Того же можно добиться командой `git reset`.

git restore: HEAD → Индекс + Рабочая область

Чтобы восстановить содержимое файла в индексе и в рабочей области (копирует содержимое из `HEAD` или `--source`):

```
$ git restore --staged --worktree README.txt
```

Того же можно добиться командой `git checkout`.

Уровни изменений

Изменения могут быть сделаны на уровне:

- всего коммита
- отдельного файла
- фрагмента файла

Изменения на уровне коммита

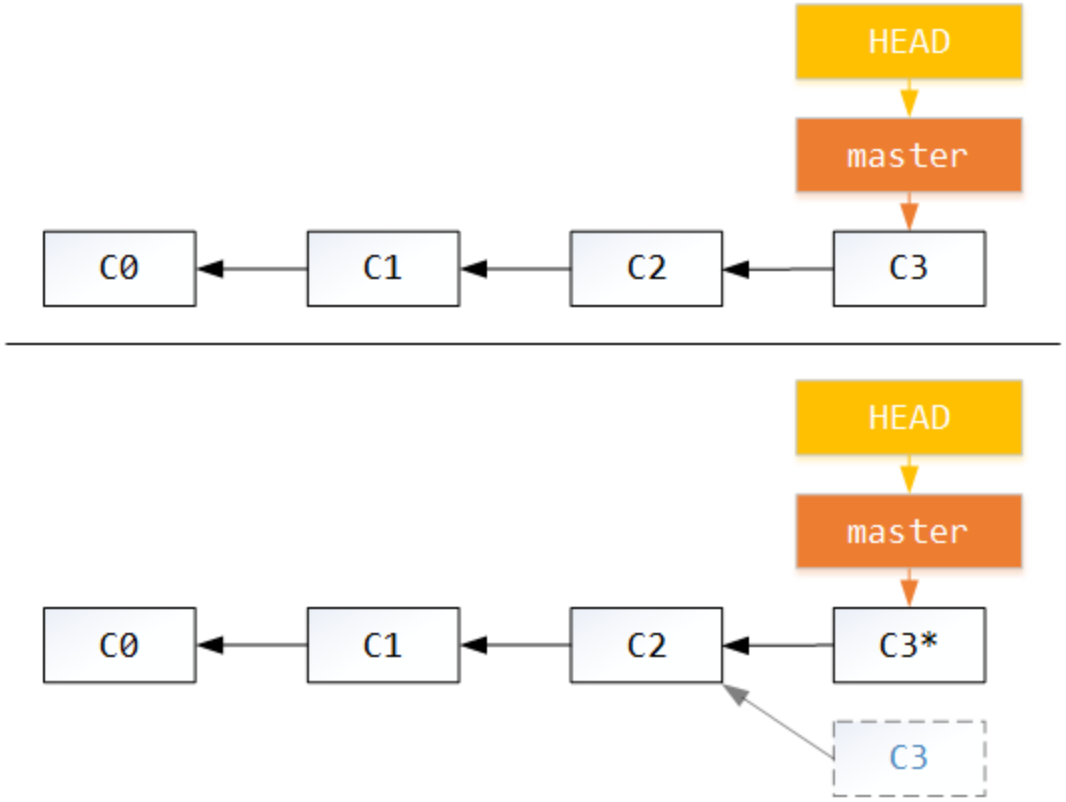
- Изменить последний коммит
- Переключиться на другой коммит
- Возврат всего проекта к определённому состоянию
- Убрать изменения в рабочем каталоге
- Убрать изменения в индексе
- Удалить коммиты
- Сплющить коммиты
- Создать коммит противоположный заданному

Изменить последний коммит

- Команда `git commit --amend` позволяет обновить последний коммит добавив изменения из текущего индекса.
- Опция `--no-edit` позволяет пропуск редактирования сообщения.

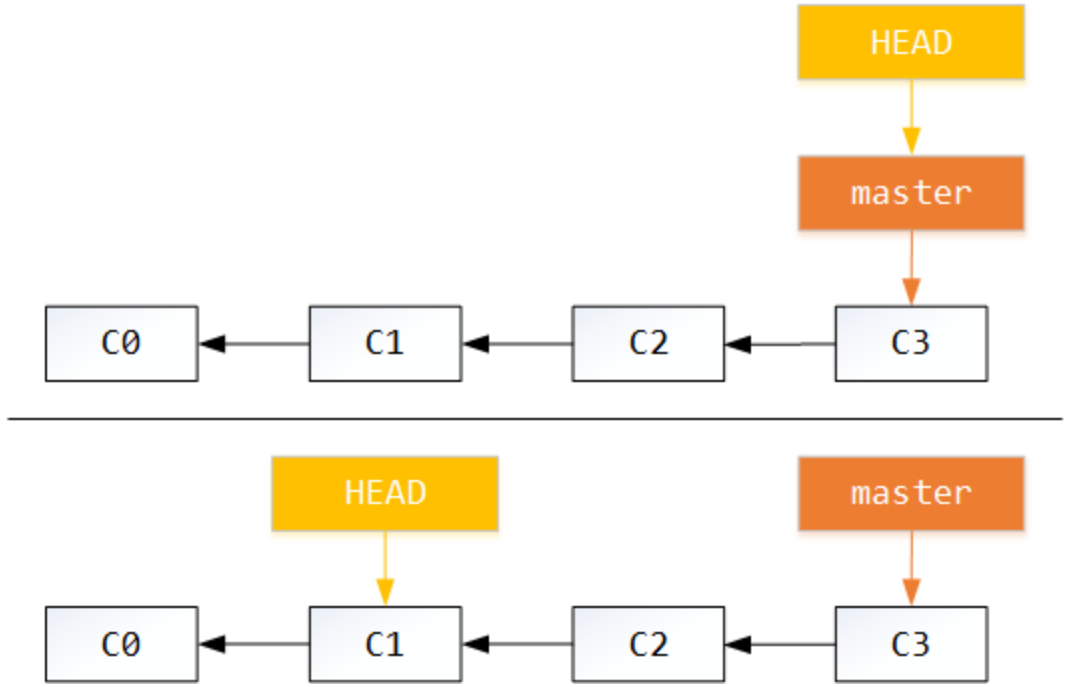
```
$ git commit -m 'initial commit'
$ git add forgotten_file

$ git commit --amend
$ git commit --amend -m "an updated commit message"
$ git commit --amend --no-edit
$ git commit --amend --reset-author
```

Изменить последний коммит (продолжение)



Переключиться на другой коммит

```
$ git checkout <commit>
```



Возврат всего проекта к определённому состоянию

```
$ git checkout <commit> .
```

Убрать изменения в рабочем каталоге

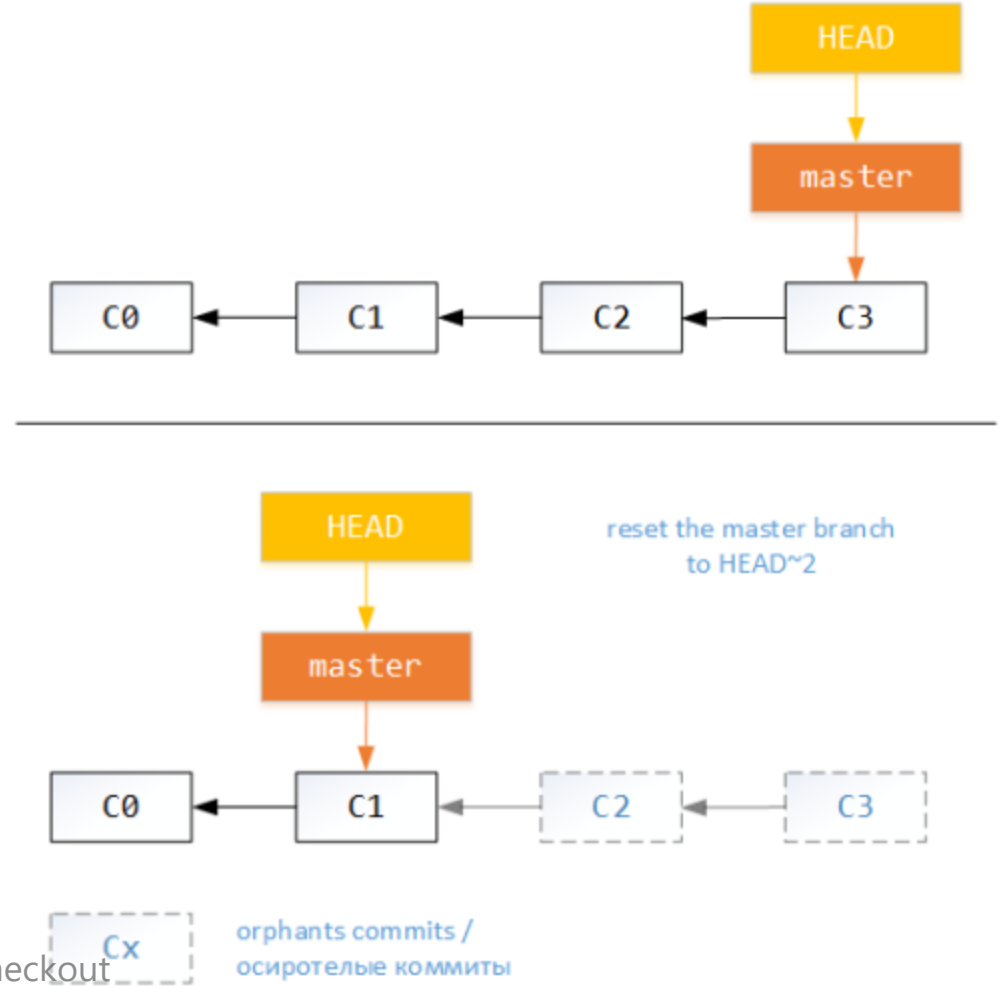
```
$ git reset --hard
$ git clean -df
```

Убрать изменения в индексе

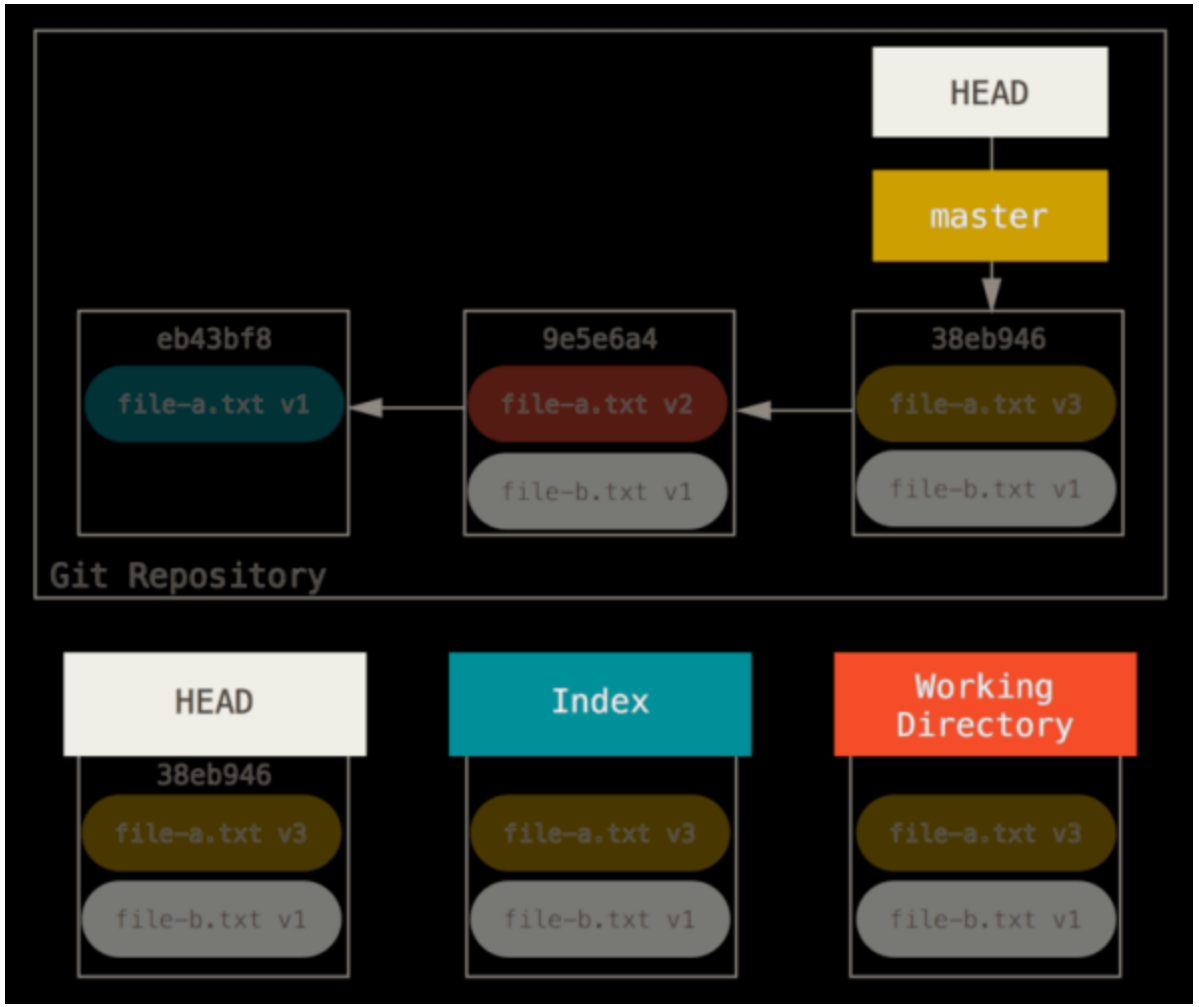
```
$ git reset --mixed HEAD
```

Удалить коммиты

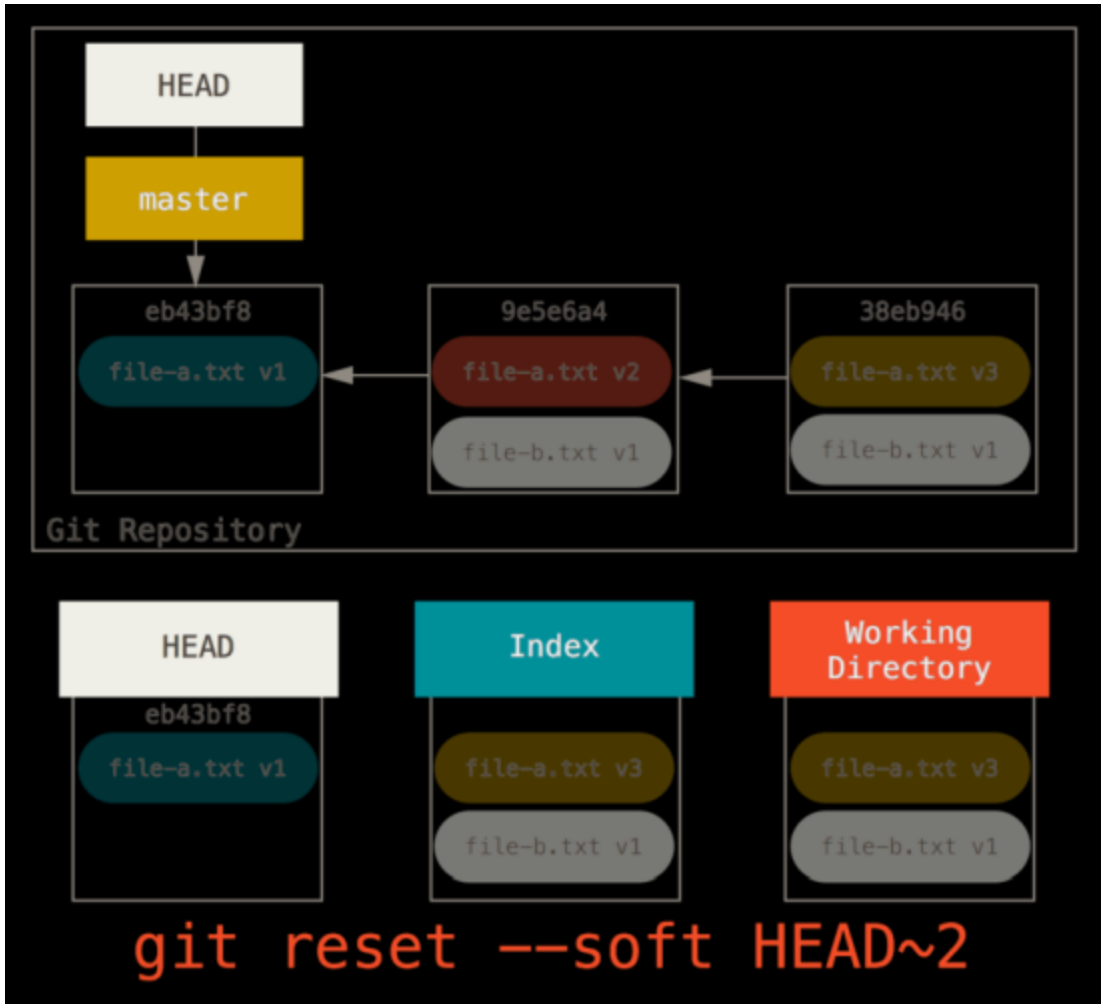
```
git checkout master
git reset HEAD~2
```



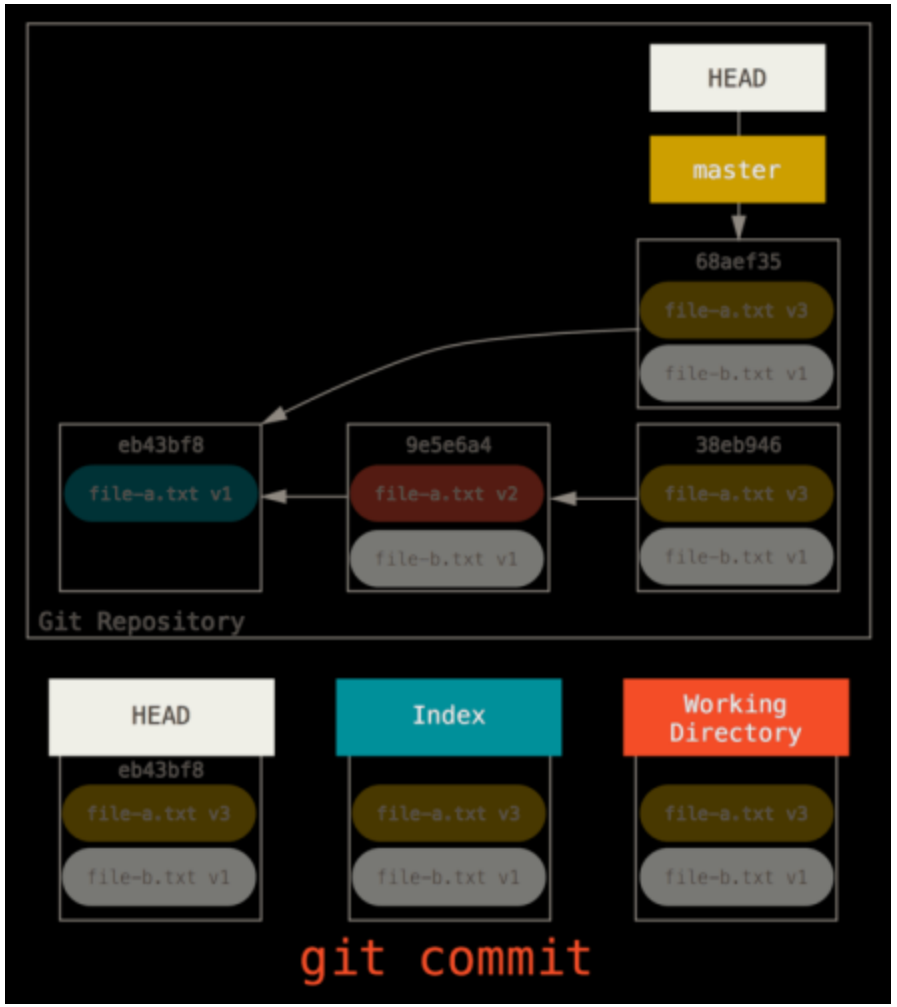
Слияние коммитов



Слияние коммитов (продолжение)



Слияние коммитов (продолжение)



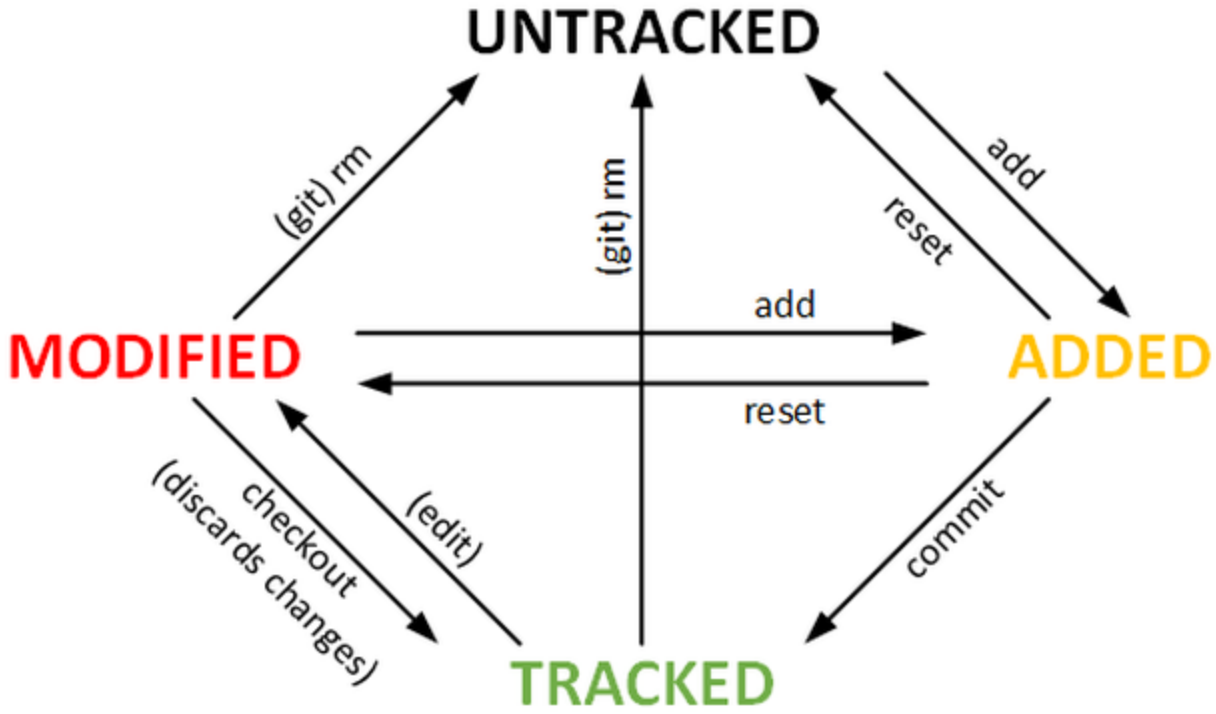
Создать коммит противоположный заданному

Команда `git revert` позволяет создать коммит противоположный указанному:

```
$ git revert 872fa7e

$ git log --oneline
e2f9a78 Revert "Try something crazy"
a1e8fb5 Make some important changes to hello.txt
872fa7e Try something crazy
435b61d Create hello.txt
9773e52 Initial import
```

Изменения на уровне файлов

Схема изменений состояния файла



Правка `(edit)` включает непосредственные изменения или получения версии файла из другого коммита.

Получение файлов для другой версии проекта

```
$ git checkout <commit> <paths>
```

Отмена `git add`

```
$ git reset HEAD <file>
```

```
$ git add .
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   README.md
```

Отмена изменения измененного файла

```
$ git checkout -- <file>
```

```
$ git status
...
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:   README.md
```

Удаление файлов из проекта

```
$ rm PROJECTS.md
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)
        deleted:    PROJECTS.md
no changes added to commit (use "git add" and/or "git commit -a")
```

```
$ git rm PROJECTS.md
rm 'PROJECTS.md'
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
    deleted:    PROJECTS.md
```

Перемещение файлов

```
$ git mv file_from file_to
```

```
$ git mv README.md README
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    renamed:    README.md -> README
```

```
$ mv README.md README
$ git rm README.md
$ git add README
```

Изменения на уровне фрагмента файла

- Частично индексировать файл

  `git add -p`

- Частично откатить изменения в файле

  `git reset -p` и `git checkout -p`

```
Apply this hunk to index and worktree [y,n,q,a,d,e,?]? ?
y - apply this hunk to index and worktree
n - do not apply this hunk to index and worktree
q - quit; do not apply this hunk or any of the remaining ones
a - apply this hunk and all later hunks in the file
d - do not apply this hunk or any of the later hunks in the file
e - manually edit the current hunk
? - print help
```

Заключение

- Нет выделенной команды `undo`.
- Нужно выполнить команду с противоположным действием.
- Изменения можно откатывать на разном уровне:
  - коммита
  - файла (-ов)
  - фрагмента файла