Git

Система контроля версий - установка, основы, состояния, ветвление, интерфейсы и хранилища

< Teach
Me
Skills / >

Урок

< Teach
Me
Skills / >

Git

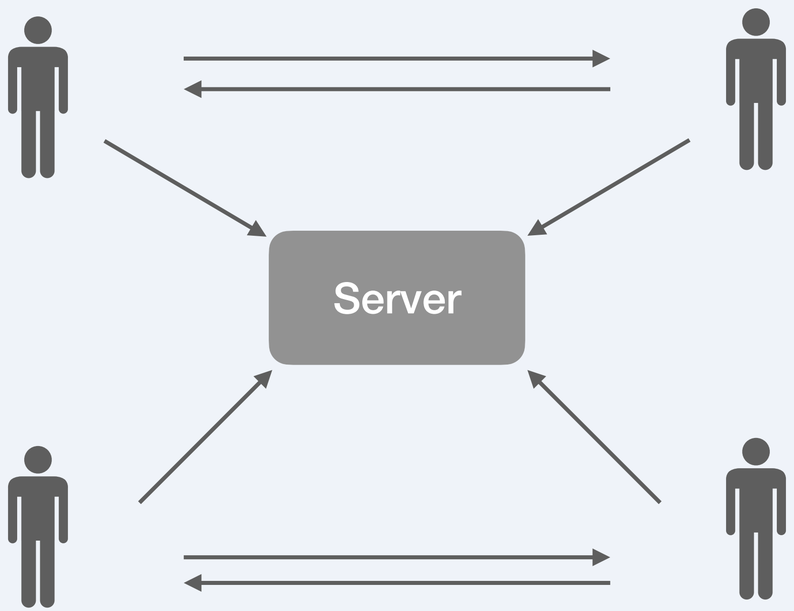
Что это такое?

Для чего нужен ?

Как им пользоваться ?

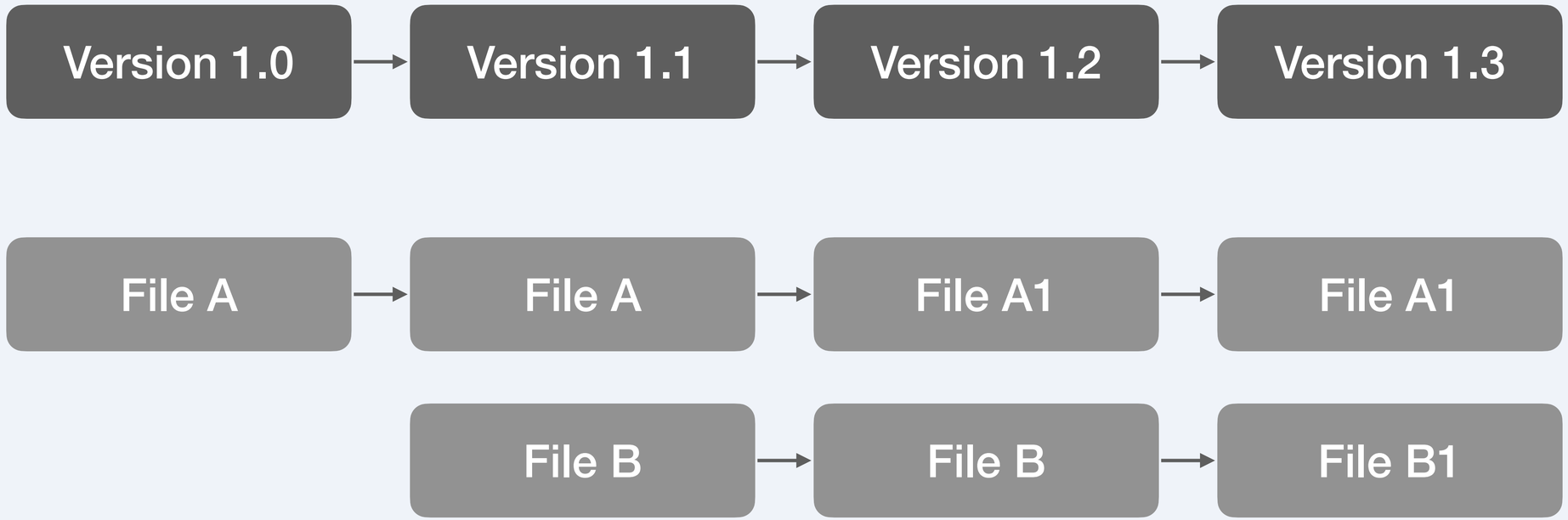
< Teach
Me
Skills / >

Git распределенная система контроля версий



< Teach
Me
Skills / >

Git отслеживает изменения в проекте



Установка Git

Загружаем и устанавливаем git → https://git-scm.com/

Графические интерфейсы git

- Github Desktop
- Git Kraken
- Sourcetree
- And other

< Teach
Me
Skills / >

Сервисы для хранения проектов

- Github
- Gitlab
- Bitbucket

Что такое репозиторий?

В репозитории хранятся все документы вместе с историей их изменения и другой служебной информацией.

То есть, эта папка вашего проекта

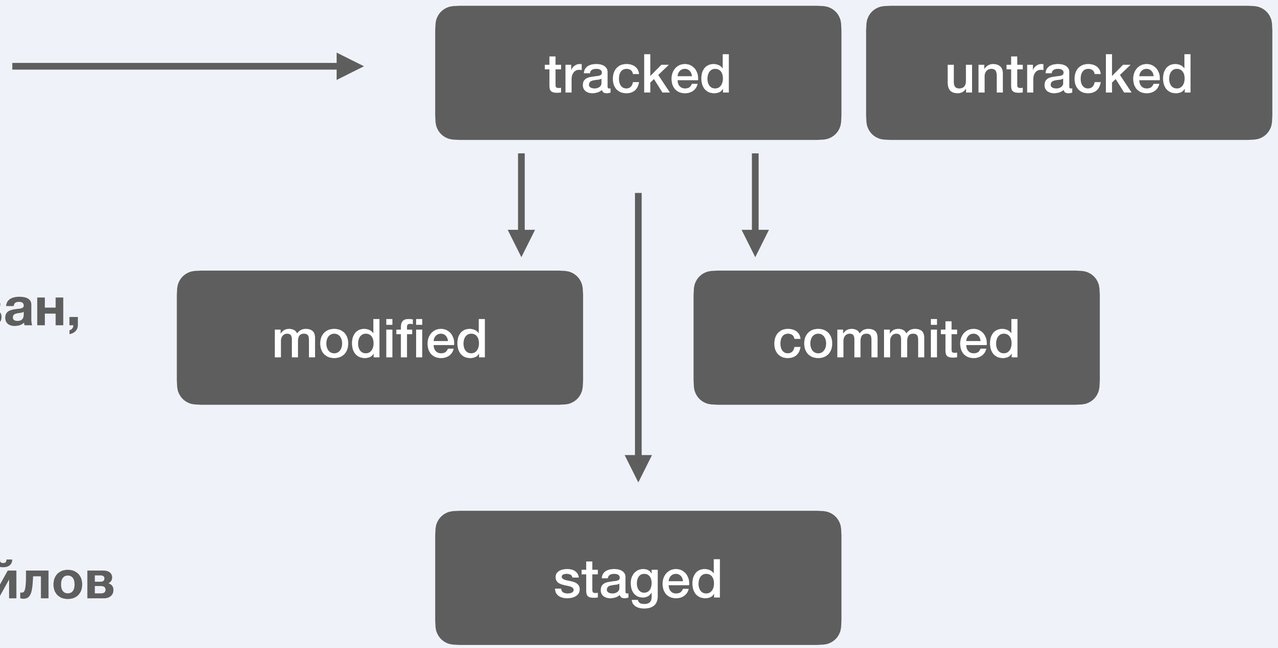
Основные команды

git init	Создает в текущей директории новую поддиректорию .git Она содержит в себе структуру git-репозитория
git status	Команда позволяющая отслеживать состояние в текущем репозитории
git add	Команда добавляет файл в индекс для фиксации его изменений
git commit	Описание сохранения подготовленных и измененных файлов
git push	Отправляет изменения в удаленный репозиторий

< Teach Me Skills / >

Состояние файлов в git

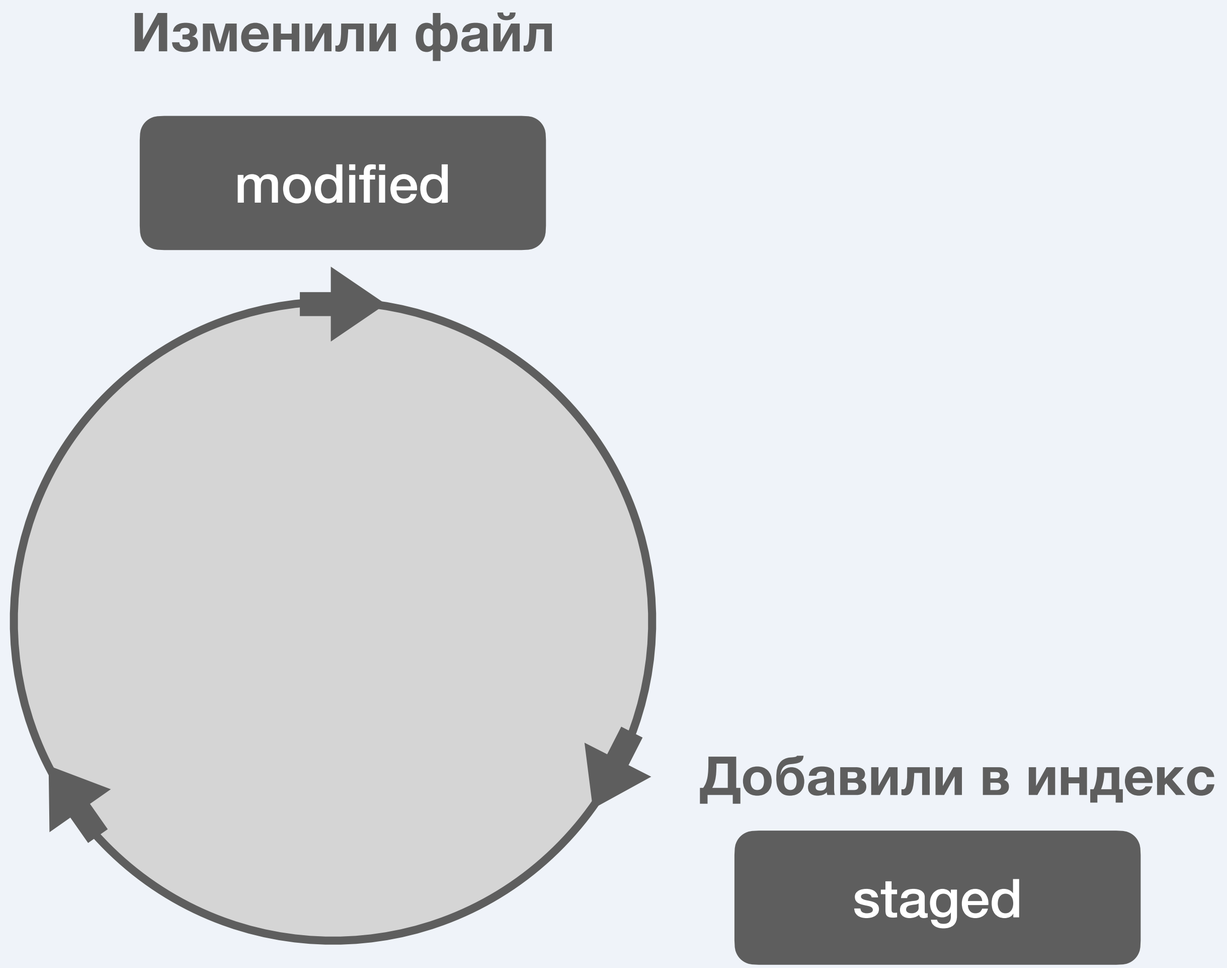
Отслеживает или нет изменения файлов



Файл изменен и не фиксирован, файл попал в commit

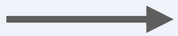
Область подготовленных файлов

Круговорот состояний



Игнорируем файлы для отслеживания

Создаем файл .gitignore

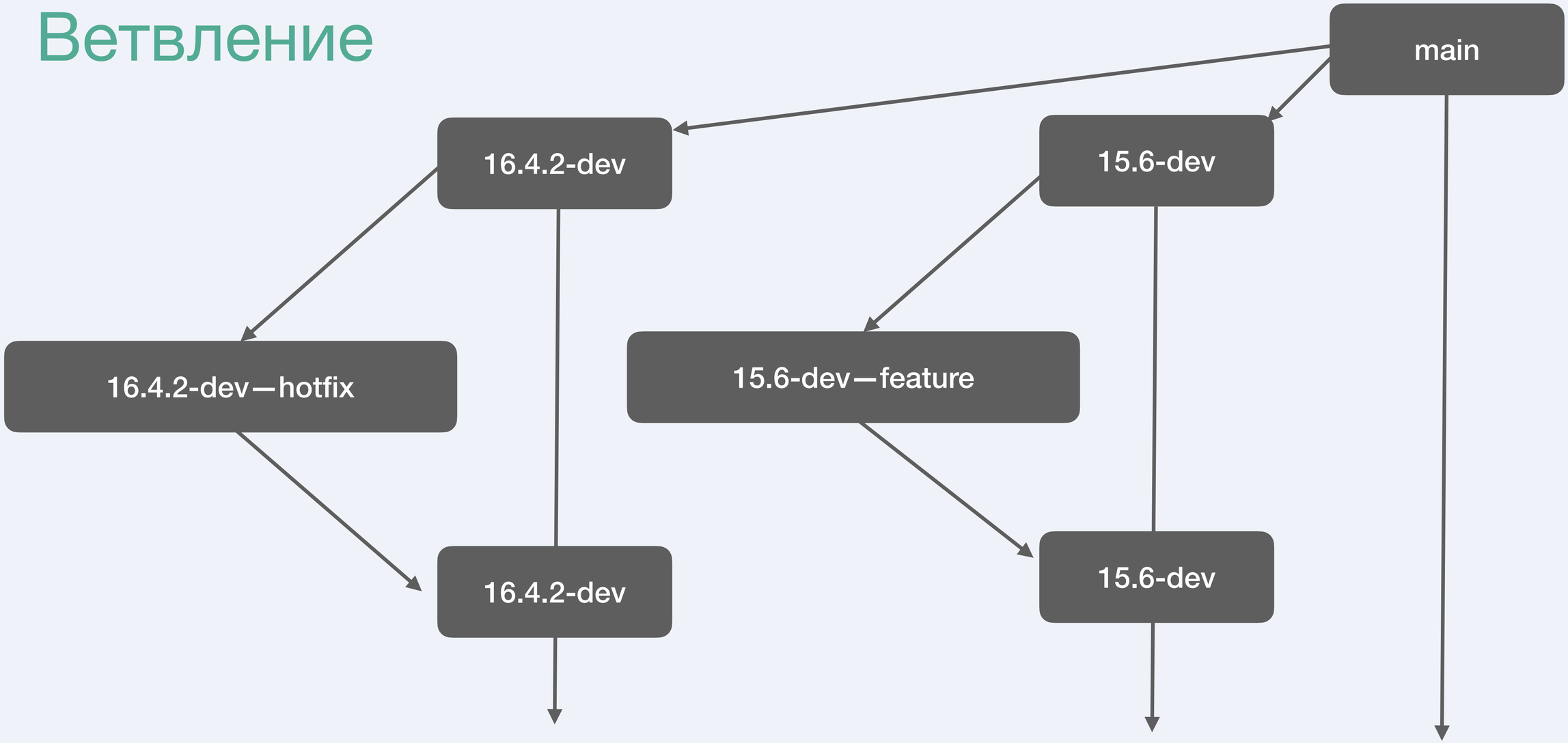


```
# Comments

*.js

/cache

form.html

log/

log/logs.txt
```

Ветвление

- **Ветка в Git это подвижный указатель на один из коммитов.**
- **Обычно ветка указывает на последний коммит в цепочке коммитов.**
- **Ветка берет свое начало от какого-то одного коммита.**

Ветвление



Основные команды

git branch	Список, создание или удаление веток
git checkout	Переключаемся между ветками
git merge	Объединяет ветки вместе
git pull	Получает и интегрирует данные с другой ветки
git clone	Получить копию существующего git репозитория

< Teach
Me
Skills / >

Дополнительные полезные команды

```
git config --global user.name "John Doe"
git config --global user.email johndoe@example.com
```

Первое, что вам следует сделать после установки Git — указать ваше имя и адрес электронной почты. Это важно, потому что каждый коммит в Git содержит эту информацию, и она включена в коммиты, передаваемые вами

Дополнительные полезные команды

git config --list --show-origin

Посмотреть все установленные настройки и узнать где именно они заданы

Дополнительные полезные команды

```
git reset --hard a3775a5485af0af20375cedf46112db5f813322a
git push --force
```

Откатываемся на предыдущий коммит

Дополнительные полезные команды

git config --global init.defaultBranch main

Когда вы инициализируете репозиторий командой git init, Git создаёт ветку с именем master по умолчанию. Начиная с версии 2.28, вы можете задать другое имя для создания ветки по умолчанию. Например, чтобы установить имя main для вашей ветки по умолчанию, выполните следующую команду

Дополнительные полезные команды

git config --list

Если вы хотите проверить используемую конфигурацию, можете использовать команду git config --list, чтобы показать все настройки, которые Git найдёт

< Teach
Me
Skills / >

Дополнительные полезные команды

git help <команда>

Например, так можно открыть руководство по команде

Дополнительные полезные команды

git clone https://github.com/libgit2/libgit2

Клонирование репозитория осуществляется командой git clone <url>. Например, если вы хотите клонировать библиотеку libgit2, вы можете сделать это следующим образом

Дополнительные полезные команды

git commit -a -m 'Add new benchmarks'

Если у вас есть желание пропустить этап индексирования, Git предоставляет простой способ. Добавление параметра -a в команду git commit заставляет Git автоматически индексировать каждый уже отслеживаемый на момент коммита файл, позволяя вам обойтись без git add

Дополнительные полезные команды

Для того чтобы удалить файл из Git, вам необходимо удалить его из отслеживаемых файлов (точнее, удалить его из вашего индекса) а затем выполнить коммит. Это позволяет сделать команда git rm, которая также удаляет файл из вашего рабочего каталога, так что в следующий раз вы не увидите его как «неотслеживаемый». Если вы просто удалите файл из своего рабочего каталога, он будет показан в секции «Changes not staged for commit» (измененные, но не проиндексированные) вывода команды git status

Дополнительные полезные команды

git rm --cached README

вы можете захотеть оставить файл на жёстком диске, но перестать отслеживать изменения в нём. Это особенно полезно, если вы забыли добавить что-то в файл .gitignore и по ошибке проиндексировали, например, большой файл с логами, или кучу промежуточных файлов компиляции. Чтобы сделать это, используйте опцию --cached

< Teach
Me
Skills / >

Дополнительные полезные команды

После того, как вы создали несколько коммитов или же клонировали репозиторий с уже существующей историей коммитов, вероятно вам понадобится возможность посмотреть что было сделано — историю коммитов. Одним из основных и наиболее мощных инструментов для этого является команда git log.

Дополнительные полезные команды

git log -S function_name

Следующим действительно полезным фильтром является опция -S, которая принимает аргумент в виде строки и показывает только те коммиты, в которых изменение в коде повлекло за собой добавление или удаление этой строки. Например, если вы хотите найти последний коммит, который добавил или удалил вызов определенной функции, вы можете запустить команду

Дополнительные полезные команды

git commit --amend

Если вы хотите переделать коммит — внесите необходимые изменения, добавьте их в индекс и сделайте коммит ещё раз, указав параметр —amend. Очень важно понимать, что когда вы вносите правки в последний коммит, вы не столько исправляете его, сколько заменяете новым, который полностью его перезаписывает.

< Teach
Me
Skills / >

Дополнительные полезные команды

git reset HEAD test.md

Например, вы изменили два файла и хотите добавить их в разные коммиты, но случайно выполнили команду git add * и добавили в индекс оба. Как исключить из индекса один из них? Команда git status напомнит вам