

# Git

**Git** — это распределённая система управления версиями, разработанная Линусом Торвальдсом в 2005 году. Является свободным программным обеспечением. Система представляет набор программ, работающих с готовыми сценариями, это позволяет создавать программы, пользовательские интерфейсы для работы с git. Git поддерживает разделение и слияние версий, включает инструменты для просмотра изменений истории проекта. Git предоставляет каждому разработчику локальную копию всей истории разработки.

Создаём новую папку:

```
~]$  
~]$ mkdir /home/denis/gitFolder  
~]$
```

Открываем созданную папку и вводим команду:

```
gitFolder]$ git init  
Инициализирован пустой репозиторий Git в /home/denis/gitFolder/.git/
```

**Git init** - создаёт в текущем каталоге новый подкаталог с именем .git, содержащий все необходимые файлы репозитория — структуру Git репозитория.

Для работы с git создаём файл и добавляем его в контроль версий:

```
gitFolder]$ touch newfile.txt  
gitFolder]$ ls -a  
.. .git newfile.txt  
gitFolder]$ git add newfile.txt
```

**Git add** добавляет файлы рабочего каталога в индекс для последующего коммита. Индекс в Git — это специальная промежуточная область, в которой хранятся изменения файлов на пути от рабочей директории до репозитория. При выполнении коммита в него попадают только те изменения, которые были добавлены в индекс.

Для того, что бы проверить в каком состоянии находится наш файл ввожу команду:

```
gitFolder]$ git status  
Изменения, которые будут включены в коммит:  
(используйте «git rm --cached <файл>...», чтобы убрать из индекса)  
новый файл:   newfile.txt
```

**git status** - Команда git status показывает состояния файлов в рабочем каталоге и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе. Вдобавок к этому выводятся подсказки о том, как изменить состояние файлов.

Можем заметить, что наш файл индексируется. Сделаем commit для сохранения изменений в нашем проекте. Но в начале введём наше имя и электронную почту потому, что каждый commit содержит эту информацию. Она включена в коммиты и не может быть изменена.

```
gitFolder]$ git config user.name "Denis Negura"  
gitFolder]$ git config user.email "bulidog.pampa16@gmail.com"
```

```
gitFolder]$ git commit -am 'new file in the project!'  
[master (корневой коммит) 7f1e17a] new file in the project!  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 newfile.txt
```

**Git commit** - Команда берёт все данные, добавленные в индекс с помощью git add, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

**git commit -a** - Выполнение коммита состояния со всеми изменениями в рабочем каталоге. Эта команда включает только изменения отслеживаемых файлов (тех, которые были в какой-то момент добавлены в историю с помощью команды git add).

**git commit -m** - Быстрая команда, которая создает коммит с указанным комментарием. По умолчанию команда git commit открывает локально настроенный текстовый редактор с предложением ввести комментарий к коммиту. При передаче параметра -m текстовый редактор не открывается, а используется подставленный комментарий.

Для просмотра журнала коммитов используем команду:

```
gitFolder]$ git log  
commit 7f1e17a75edcbb546e5cfec7edc8415087b9a9fb (HEAD -> master)  
Author: Denis Negura <bulidog.pampa16@gmail.com>  
Date: Mon Feb 7 12:19:54 2022 +0200  
  
    new file in the project!
```

**Git log** - По умолчанию (без аргументов) git log перечисляет коммиты, сделанные в репозитории в обратном к хронологическому порядку — последние коммиты находятся вверху. Из примера можно увидеть, что данная команда перечисляет коммиты с их SHA-1 контрольными суммами, именем и электронной почтой автора, датой создания и сообщением коммита.

Коммит был создан в ветке «master», на него ссылается ссылка HEAD. Введём данные в наш файл для того, чтобы определить работу коммитов.

```
gitFolder]$ echo "git" >> newfile.txt
gitFolder]$ cat newfile.txt
git
```

Используем команду git status для вывода состояния нашего файла:

```
На ветке master
Изменения, которые не в индексе для коммита:
  (используйте «git add <файл>...», чтобы добавить файл в индекс)
  (используйте «git restore <файл>...», чтобы отменить изменения в рабочем каталоге)
        изменено:      newfile.txt

нет изменений добавленных для коммита
(используйте «git add» u/или «git commit -a»)
```

Файл был изменён. Создаём новый коммит для сохранения изменений:

```
gitFolder]$ git commit -am 'New data in the file'
[master 837f26b] New data in the file
1 file changed, 1 insertion(+)
```

Командой git log наблюдаем следующие изменения:

```
commit 837f26b54f5629110dc5d09779f0e7b3ef1d9b36 (HEAD -> master)
Author: Denis Negura <bulidog.pampa16@gmail.com>
Date:   Mon Feb 7 12:57:23 2022 +0200

    New data in the file

diff --git a/newfile.txt b/newfile.txt
index e69de29..5664e30 100644
--- a/newfile.txt
+++ b/newfile.txt
@@ -0,0 +1 @@
+git

commit 7f1e17a75edcbb546e5cfec7edc8415087b9a9fb
Author: Denis Negura <bulidog.pampa16@gmail.com>
Date:   Mon Feb 7 12:19:54 2022 +0200

    new file in the project!

diff --git a/newfile.txt b/newfile.txt
new file mode 100644
index 0000000..e69de29
```

Ссылка HEAD сейчас указывает на последний коммит. Вернёмся к предпоследнему коммиту командой:

```
gitFolder]$ git revert HEAD
```

**Git reset** - используется в основном для отмены изменений. Она изменяет указатель HEAD и, опционально, состояние индекса.

Нам предложено ввести комментарий:

```
GNU nano 6.0
Revert "New data in the file"

This reverts commit 837f26b54f5629110dc5d09779f0e7b3ef1d9b36.

# Пожалуйста, введите сообщение коммита для Ваших изменений. Стр
# начинающиеся с «#» будут проигнорированы, а пустое сообщение
# отменяет процесс коммита.
```

Сохраняем файл с предложенным комментарием и возвращаемся в терминал.

```
[master 588bda7] Revert "New data in the file"
1 file changed, 1 deletion(-)
```

Работа команды завершилась. Просматриваем журнал коммитов.

```
commit 588bda75fcb907c9f39e8e2a9ee828fdc8affa6c (HEAD -> master)
Author: Denis Negura <bulidog.pampa16@gmail.com>
Date: Mon Feb 7 13:13:56 2022 +0200

    Revert "New data in the file"

    This reverts commit 837f26b54f5629110dc5d09779f0e7b3ef1d9b36.

commit 837f26b54f5629110dc5d09779f0e7b3ef1d9b36
Author: Denis Negura <bulidog.pampa16@gmail.com>
Date: Mon Feb 7 12:57:23 2022 +0200

    New data in the file

commit 7f1e17a75edcbb546e5cfec7edc8415087b9a9fb
Author: Denis Negura <bulidog.pampa16@gmail.com>
Date: Mon Feb 7 12:19:54 2022 +0200

    new file in the project!
```

Второй коммит не был стёрт, был создан новый коммит, который анулировал изменения прошлого. Следовательно, первый и последний коммиты идентичны.

Как мы можем заметить наш файл пуст.

```
gitFolder]$ cat newfile.txt  
gitFolder]$
```

Вновь используем команду `git revert` для возвращения к прошлому коммиту.

```
gitFolder]$ git revert 588bda75fcb907c9f39e8e2a9ee828fdc8affa6
```

На этот раз я использую ключ `SHA-1`. Открываем файл:

```
GNU nano 6.0  
git
```

Файл содержит введённые нами ранее изменения.