Devpractice
Разработка программного обеспечения, технологии и наука

Git для начинающих. Часть 8. Добавление, удаление и переименование файлов в репозитории

Автор: Marat Abdrakhmanov | 22.03.2018 1 комментарий

В рамках данного урока рассмотрим вопросы, касающиеся добавления, удаления и переименования файлов в *git* репозитории.

Добавление файлов в *git* репозиторий

Добавление файлов в репозиторий – это достаточно простая операция, мало чем отличающаяся от отправки изменений в отслеживаемых файлах в репозиторий. Мы уже не раз выполняли эту операцию в предыдущих уроках, но сделаем это ещё раз. Создадим новый репозиторий, для этого перейдите в каталог, в котором вы хотите его расположить и введите команду *git init*.

```
> git init
```

Создайте в каталоге файл *README.md* любым удобным для вас способом, мы сделаем это с помощью команды *touch*.

```
> touch README.md
```

Теперь проверим состояние отслеживаемой директории.

```
> git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        README.md

nothing added to commit but untracked files present (use "git add" to track)
```

Как вы можете видеть: в рабочей директории есть один неотслеживаемый файл *README.md*. *Git* нам подсказывает, что нужно сделать для того, чтобы начать отслеживать изменения в файле *README.md*: необходимо выполнить команду *git add*, сделаем это.

```
> git add README.md
```

Посмотрим ещё раз на состояние.

```
> git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   README.md
```

Видно, что информация о появлении нового файла попала в *stage*. Для того чтобы это изменение зафиксировалось в репозитории необходимо выполнить команду *git commit*.

```
> git commit -m "add README.md file"
[master (root-commit) 0bb6c94] add README.md file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 README.md
```

Теперь в рабочей директории и в *stage* нет объектов, информацию об изменении которых необходимо внести в репозиторий.

```
> git status
On branch master
nothing to commit, working tree clean
```

В репозиторий был сделан один коммит.

```
> git log --oneline
0bb6c94 add README.md file
```

Удаление файлов из *git* репозитория и из *stage*

Удаление файла из *stage*

Вначале разберемся со *stage*. Создадим ещё один файл.

```
> touch main.c
```

“Отправим” файл *main.c* в *stage*.

```
> git add main.c
```

Внесем изменения в *README.md*.

```
> echo "# README" > README.md
```

Информацию об этом также отправим в *stage*.

```
> git add README.md
```

Посмотрим на состояние *stage*.

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

        modified:   README.md
        new file:   main.c
```

Если нам необходимо убрать из *stage*, какой-то из этих файлов (main.c или README.md), то для этого можно воспользоваться командой *git –rm cashed <filename>*, сделаем это для файла *main.c*.

```
> git rm --cached main.c
rm 'main.c'
```

Теперь посмотрим на состояние рабочей директории и *stage*.

```
> git status
On branch master
Changes to be committed:
```

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

        modified:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        main.c
```

Видно, что изменения в файле *README.md* готовы для коммита, а вот файл *main.c* перешел в состояние – неотслеживаемый. Отправим *main.c* в stage и, после этого, сделаем коммит в репозиторий.

```
> git add main.c
> git commit -m "add main.c and do some changes in README.md"
[master 49049bc] add main.c and do some changes in README.md
 2 files changed, 1 insertion(+)
 create mode 100644 main.c
```

Удаление файлов из *git* репозитория

Удалить файл из репозитория можно двумя способами: **первый** – удалить его из рабочей директории и уведомить об этом *git*; **второй** – воспользоваться средствами *git*. **Начнем с первого способа**. Для начала посмотрим, какие файлы у нас хранятся в репозитории.

```
> git ls-tree master
100644 blob 7e59600739c96546163833214c36459e324bad0a    README.md
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    main.c
```

Удалим файл *main.c* из рабочей директории.

```
> rm main.c
> ls
README.md
```

Уведомим об этом систему *git*.

```
> git rm main.c
rm 'main.c'
```

Вместо команды *git rm* можно использовать *git add*, но само слово *add* в данном случае будет звучать несколько неоднозначно, поэтому лучше использовать *rm*. На данном этапе еще можно вернуть все назад с помощью команды *git checkout — <filename>*, в результате, в нашу рабочую директорию будет скопирован файл из репозитория. Создадим коммит, фиксирующий удаление файла.

```
> git commit -m "remove main.c"
[master d4e22ae] remove main.c
```

```
 1 file changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 main.c
```

Теперь в репозитории остался только один файл *README.md*.

```
> git ls-tree master
100644 blob 7e59600739c96546163833214c36459e324bad0a    README.md
```

Второй способ – это сразу использовать команду *git rm* без предварительного удаления файла из директории. Вновь создадим файл *main.c* и добавим его в репозиторий.

```
> touch main.c
> git add main.c
> git commit -m "add main.c file"
[master 6d93049] add main.c file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 main.c
> git ls-tree master
100644 blob 7e59600739c96546163833214c36459e324bad0a    README.md
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    main.c
```

Удалим файл из репозитория.

```
> git rm main.c
rm 'main.c'

> git commit -m "deleted: main.c file"
[master ba7d027] deleted: main.c file
 1 file changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 main.c
```

Файла *main.c* больше нет в репозитории.

```
> git ls-tree master
100644 blob 7e59600739c96546163833214c36459e324bad0a    README.md
```

Его также нет и в рабочем каталоге.

```
> ls
README.md
```

Удалите файл *README.md* из репозитория самостоятельно.

Переименование файлов в *git* репозитории

Как и в случае с удалением, переименовать файл в *git* репозитории можно двумя способами – с использованием и без использования средств операционной системы.

Первый способ. Создадим файл *test_main_file.c* и добавим его в репозиторий.

```
> touch test_main_file.c

> git add test_main_file.c

> git commit -m "add test_main_file.c"
[master 6cf53ac] add test_main_file.c
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_main_file.c
```

Содержимое репозитория после этого будет выглядеть так.

```
> git ls-tree master
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    test_main_file.c
```

Переименуем его на *test_main.c*.

Сделаем это в рабочей директории.

```
> mv test_main_file.c test_main.c
```

Теперь отправим изменение в репозиторий.

```
> git add .
> git commit -m "Rename test_main_file.c"
[master 79528c4] Rename test_main_file.c
 1 file changed, 0 insertions(+), 0 deletions(-)
 rename test_main_file.c => test_main.c (100%)
```

В репозитории и в рабочей директории будет находится только файл *test_main.c*.

```
> git ls-tree master
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    test_main.c

> ls
test_main.c
```

Второй способ.

В рамках второго способа рассмотрим работу с командой *git mv*. Переименуем файл *test_main.c* в *main.c*. Текущее содержимое репозитория и рабочего каталога.

```
> git ls-tree master
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    test_main.c
> ls
test_main.c
```

Переименуем файл *test_main.c* на *main.c* средствами *git*.

```
> git mv test_main.c main.c

> git commit -m "Rename test_main.c file"
[master c566f0e] Rename test_main.c file
 1 file changed, 0 insertions(+), 0 deletions(-)
 rename test_main.c => main.c (100%)
```

Имя файла изменилось как в репозитории так и в рабочем каталоге.

```
> git ls-tree master
100644 blob e69de29bb2d1d6434b8b29ae775ad8c2e48c5391    main.c

> ls
main.c
```

> *Отличный курс по git делают ребята из **GeekBrains**, найдите в разделе "Курсы" курс "Git. Быстрый старт", он **бесплатный**!*

<<<Часть 7. Поговорим о *HEAD* и *tree-ish* Часть 9. Как удалить коммит в git?>>>

Раздел: Git Git для начинающих Метки: Git, Git для начинающих, Уроки по Git

Git для начинающих. Часть 8. Добавление, удаление и переименование файлов в репозитории: 1 комментарий

Игорь
09.12.2021

В целом, очень неплохо написанный начальный курс. Что бросилось в глаза, так это то, что вы вводите такую конструкцию как

git rm –cached

без объяснений, что это значит и чем отличается от простого git rm. Между тем это достаточно важный и не слишком очевидный момент (cached, staged). Я не искал по всем частям опубликованного вами материала. Возможно, есть и другие подобные места, требующие чуть более основательных пояснений. Я понимаю, что в короткую статью очень сложно втиснуть все нужное, не раздувая ее до объема книги, но тем не менее.