|  |
| --- |
| **cd ->** выйти в корневую директорию пользователя |
| **cd d: ->**  перейти в диск **d** |
| **cd d:/my-example –>** перейти в директорию **my-example** |
| **cd .. ->** перейти из директории на уровень выше |
| **mkdir log ->** создание папки с именем log |
| **rmdit log** -**>** удаление папки с именем log |
| **rm file1.txt ->** удаление файла file1.txt |
| **touch file1.txt ->** создание файла file1.txt |
| **echo “some text” > file1.txt ->** создать текстовый файл file1.txt с содержимым в кавычках |
| **nano file2.txt ->** внестиизменения в файл **file2.txt** если его нет, то создать файл **file2.txt** и прописать содержимое |
| **cat file1.txt ->** показывает содержимое файла **file1.txt** |
|  |
| git config [--**global**] user.name "Your Name"  git config [--**global**] user.email "you@example.com" |
| git config user.email "you@example.com"  git config user.name "Your Name"  git config user.password "your password" |
| **git config --list --global ->** покажет **user.name** и **user.email** |
| **git --version ->** просмотр версии гита |
| **git init . ->** инициализирует новый репозиторий в текущей директории |
| **git init my\_project ->** создаёт директорию my\_project и сразу инициализирует в ней репозиторий |
| **git status ->** отображает список измененных, добавленных и удаленных файлов |
| **git add ->** добавляет указанные файлы в индекс |
| **git add \* либо git add . ->** добавляет все файлы в индекс |
| **git reset ->** отменяет действие команды **git add** на файл |
| **git commit –a –m ‘some text’ ->** индексируетотслеживаемыефайлы и сразу создаёт коммит |
| **git commit -m “some text” –>** фиксирует добавленные в индекс изменения с определенным сообщением |
| **git push ->** отправляет изменения на удаленный репозиторий |
| **find .git/objects ->** показывает содержимое директории **objects** |
| **git cat-file -p [хэш] ->** выводит содержимое объекта с данным хэшом |
| **git cat-file -p master^{tree} ->** список файлов с их хэшами  (master^{tree} указывает на дерево, соответствующее последнему коммиту ветки **master**) |
| **git log ->** история изменений (коммитов) (выйти **q**) |
| **git log -2 ->** последние 2 коммита (-1 - последний коммит) |
| **git log -1 -p ->** покажет разницу внесённую в каждый коммит |
| **git log --stat ->** покажет статистику изменённых файлов для каждого коммита (**+/- changed, insertions**) |
| **git log --oneline ->** выводит коммиты в одну строку, показывая только хэш коммита и commit message |
| **git log --pretty=format:”%H %t %an” ->** формат вывода лога |
| Другие опции для команды git log можно посмотреть в руководстве по использованию Git:  <https://git-scm.com/book/ru/v2/Основы-Git-Просмотр-истории-коммитов#rlog_options> |
|  |
|  |
|  |
| **git checkout -- file1.txt ->** отменяет изменения в рабочем каталоге, переключает на указанный файл |
| **git checkout хэш-коммита ->** переводит на любой коммит в истории наших изменений |
| **git diff --staged ->** выводит разницу между индексом и последним коммитом |
| **git show ->** показывает информацию по отдельному коммиту |
| **git commit --amend ->** редактор коммита |
| **git commit --amend -m “some text” ->** замена коммита без редактора |
| **git reset ->** позволяет удалять коммиты |
| **git reset --hard HEAD~2** **->** используется для удаления двух последних коммитов |
| **git clean ->** используется для операций с не отслеживаемы-ми файлами |
| **git revert ->** используется для обращения изменений, внесенных в коммит из истории проекта |
| **git clone** [**https://github.com/kotBantik/mysuperproject1**](https://github.com/kotBantik/mysuperproject1) **->** клонирует удаленный репозиторий на локальную машину |
| **git remote add origin** [**https://github.com/KotBantik/something.git**](https://github.com/KotBantik/something.git)  **->** связывает наш локальный репозиторий с удаленным репозиторием |
| **git push origin ->** отправляет коммиты на сервер |
| **git remote add origin HEAD ->** отправляет на сервер текущую ветку со всеми коммитами, которые мы сделали в ней. |
| **git checkout -b имя\_ветки ->** создаём новую ветку и одновременно переключаемся на неё |
| **git checkout имя\_ветки ->** переключает на локальную ветку |
| **git stash ->** откладывает изменения, не добавляя их в коммит |
| **git stash list** **->** показывает список всех отложенных изменений |
| **git stash pop ->** забирает данные из stash и удаляет их |
| **git stash apply ->** забирает данные из stash, но не удаляет их |
| **git stash clear ->** удаляет все файлы, которые находятся в stash |
| **git branch ->** по умолчанию выводит список локальных веток. С ключами **-r**, **-a** можно вывести, соответственно, либо только удаленные ветки, либо все ветки. |
| **git branch -d имя\_ветки ->** удаляет ветку, которая полностью слита с какой-то другой веткой |
| **git branch -D имя\_ветки ->** удаляет любую локальную ветку |
| **git push origin --delete имя\_ветки ->** удаляет ветку с сервера |
| **git push origin -d имя\_ветки ->** удаляет ветку с сервера |
| **git fetch ->** стягивает информацию о ветках на сервере к нам на локальную машину. |
| **git clean -f ->** удаляет все неотслеживаемые файлы |
| **git clean -fd** **->** удаляет неотслеживаемые файлы и папки |
| **git push -u origin имя\_ветки ->** создает в удаленном репозитории ветку, соответствующую локальной ветке и обе связывает. |
| **git push origin HEAD ->** пушитв удалённый репозиторийтекущую ветку |
| **git cherry-pick 40\_символьный\_номер\_коммита** **->** позволяет брать коммит с одной ветки и повторять его на другой ветке |
| **git blame file1.txt ->** отображение метаданных автора, связанных со строками, которые были внесены в файл при коммите. |
|  |
|  |
|  |
|  |
|  |