

git cheat sheet

create & clone

создать репозиторий в текущей директории `git init [название проекта]`
создать локальную копию удалённого репозитория `git clone /path/to/repository`

add & remove

добавить указанные файлы в индекс `git add <filename>`
индексировать все изменения в текущем каталоге `git add .`
-i - интерактивный режим
удалить файл из рабочей директории `git rm <filename>`
удалить файл из-под контроля версий `git rm -cached <filename>`
файл не удаляется с диска
переименовать/переместить файл `git mv <lastname> <newname>/<directory>`
удалить неотслеживаемые (untracked) файлы из рабочей директории `git clean -fd`
f - force, d - для рекурсивного удаления из подкаталогов
посмотреть статус репозитория `git status`

commit & synchronize

коммит изменений `git commit -m "Commit message"`
add + commit `git commit -a -m "Commit message"`
не работает для добавления в индекс новых неотслеживаемых файлов
добавить изменения из индекса к последнему коммиту/исправить ошибку в описании `git commit <filename> -m "Commit message"`
`git commit --amend -m "Commit message"`
переключиться на указанный коммит `git checkout <commit_hash>`
Где я? Git branch
Вернуться обратно: git switch -
загрузить коммит на удаленный сервер `git push -u origin master`
origin - имя репозитория
master - ветка, которую мы загружаем
-u позволяет в последующем не указывать название ветки, а загружать ту ветку, на которую указывает HEAD
-force перезапишет ветку
добавить удалённый репозиторий `git remote add origin <server_link>`
показать доступные удалённые репозитории `git remote`
показать информацию об удалённом репозитории `git remote show [удалённый_репозиторий]`
отправить изменения в удалённый репозиторий `git push [удалённый_репозиторий] [ветка]`
загрузить изменения из удалённого репозитория `git pull`
pull = fetch + merge
загрузить изменения из удаленного репозитория, безопасная синхронизация `git pull --rebase = fetch + rebase`
`git fetch`

branches

создать новую ветку `git branch <branch>`
(новый указатель на текущий коммит)
переключиться на ветку `--track - отслеживать ветку`
вывести все ветки `git switch <branch>/git checkout <branch>`
`git branch`
-a список удалённых веток
создать ветку и переключиться на неё `git checkout -b <branch>`
удалить ветку `git branch -d <branch>`
загрузить ветку в удаленный репозиторий `git push origin <branch>`
перезагрузить ветку `git rebase master`
-i - интерактивный rebase
создать ветку, которая будет следить за удаленной веткой o/main `git branch -f <branch> HEAD~3`
`git checkout -b <branch> o/main`
указать ветке следить за o/main `git branch -u o/main <branch>`

merge

объединить указанную ветку с основной
отобразить неиндексированные изменения в рабочей директории
отобразить проиндексированные изменения
показать разницу между двумя ветками

```
git merge <branch>
git diff
Для перемещения вниз нажать f, для перемещения вверх - b или u. Для выхода из режима просмотра - q
git diff --staged / git diff --cached
git diff <source branch> <target branch>
```

tagging

создать новую метку
показать все теги
удалить тег
вывести историю коммитов

```
git tag <tag> <commit_id>
git tag
git tag -d <tag>
git log
--oneline одной строкой
--graph вывести график отображающий структуру ветвления
--all история коммитов по всем веткам
-p показать изменения коммитов
-n2 показать 2 последних коммита
git log -grep message
git log -- <filename>
git show <commit_id>
git blame <filename>
git grep <text>
-i - искать без учета регистра
git describe <ref>
ref - это что-либо, что указывает на конкретный коммит
Вывод команды: <tag>_<numCommits>_g<hash>, где
tag - это ближайший тег в истории изменений,
numCommits - это на сколько далеко мы от этого тега,
a hash - это хеш коммита, который описывается.
```

stash

«припрятать» изменения в рабочей директории независимо от того, добавлены они в индекс или нет
восстановить спрятанные изменения работает по принципу стека
список стэшей
удалить хранилище (стэш)

```
git stash
git stash pop
git stash list
git stash drop <stash_id>
```

restore

вернуть файл в первоначальное состояние
отменить изменения неиндексированных файлов/**восстановить** удаленный файл
отменить индексирование изменений
отменить(откатить) коммит
отменить все коммиты после заданного только для локальных репозиториев!

```
git checkout -- <filename>
git restore <filename>
git restore --staged <filename>
git revert 1b2e1d63ff
git reset <commit>
--soft-изменения остаются в индексе и рабочей директории
--mixed-изменения остаются только в рабочей директории
--hard - полное удаление, включая рабочую директорию
^ - перемещение на 1 коммит назад, т.е. найти родителя
^2 - найти второго родителя
main^^ -прародитель (родитель родителя) ветки main
HEAD указывает на последний коммит (текущий выбранный)
HEAD~ означает "один коммит от последнего"
HEAD~2 - "два последних коммита"
```

tips

встроенный в git графический интерфейс
выводить в логе коммит на одной строке

добавить алиас

копировать коммиты в HEAD
после HEAD
справка

```
gitk
git config format.pretty oneline
git log --pretty=format:"%h %ad | %s%d [%an]" --graph --date=short
git config --global alias.hist "log --pretty=format:'%h %ad | %s%d [%an]' --graph --date=short"
git cherry-pick <Commit1> <Commit2> <...>
git help
```