Git Cheat sheet

hupstream campus

Manage repositories

- `git init [--bare] <directory>`

Create a new repository

- `git clone <url> [directory]`

Get a remote repository

Manage commits

- `git add <file....>`

Add modifications to the staging area

- `git commit [-m <message>]`

Commit modifications

- `git rm <file...>`
`git commit`

Delete a file

- `git mv <src> <dst>|<dir_dst>`

Rename / move a file

- `git reset HEAD <file>`

Unstage a file

Undo commits

- `git reset --mixed <SHA1>`

Move HEAD and get all in the working directory

- `git reset --soft <SHA1>`

Move HEAD and get all in the staging area

- `git reset --hard <SHA1>`

Move HEAD and get nothing

Manage local branches

- `git branch <branch> [SHA1]`

Create a branch

- `git checkout -b <branch>`

Create a branch and change for it

- `git status`
`git checkout <branch>`

Change branch

- `git branch [-v]`

List local branches

- `git merge <branch>`

Merge branches

- `git status`
`edit file(s)`
`git add <file>`
`git commit`

Manage merge conflict

- `git mergetool -t <tool> <file>`

Edit a file during merge conflict using a gui tool

- `git merge --abort`

Cancel a merge in progress

- `git log --oneline <branch>..<branch>`

Compare 2 branches histories

- `git branch -m <old> <new>`

Rename a local branch

- `git branch -d|-D <branch>`

Delete a local branch

Get some information

- `git status`

Repository status : staging and working area, new files, cnflicts, local branch / remote branch

- `git diff [--cached]| [SHA1 SHA1]`

Display the contents of the modifications: commits, staging area, working area

- `git blame [SHA1] <file>`

List of authors who made the last modifications to each line of the file

- `git log [--oneline] [SHA1]`

Commit history of the current branch

- `git show <SHA1>:<file>`

Check a given version of a file

- `git checkout [SHA1]-- <file>`

Recover a file in the work area for a given version

- `git remote show origin`

Information about remote repository

- `git log --oneline -- <file>`

Look for commits that modify a given file

- `git log --oneline --name-status`

Commit history including list of modified files

- `git log --oneline -diff-filter=<F>`

using <F> : A added, D deleted, M modified, R renamed, C copied

Sort commits history using a filter

- `git log --oneline --grep="<string>"`

Commits history filtered using a string inside commits messages

Git Cheat sheet

hupstream campus

Manage remote branches

- `git pull [server branch]`

Update the remote references and local branches

- `git fetch`

Update remote references only

- `git push [server branch]`

Push changes to the remote server

- `git checkout <branch>`
`git checkout -b <local> origin/<dst>`

Recover a remote branch

Keep your pending modifications aside

- `git stash [save <message>]`

Put aside your work

- `git stash list`

Liste stash zones

- `git stash show -p <stash@{x}>`

Check the content of a stash zone

- `git stash apply <stash @{x}>`

Recover the content of a stash zone

- `git stash drop <stash@{x}>`

Delete a stash zone

- `git stash pop <stash@{x}>`

Recover and delete a stash zone

- `git stash branch <branch> <stash@{x}>`

Recover the content of a stash zone in a dedicated branch

Modify commits

- `git commit --amend`

Modify the last commit: commit message, commit content

- `git revert [SHA1...]`

Create a reverse commit (usable on commits already pushed to the server)

Reorganize commits, prepare push

- `git rebase <branch>`

Reorganize the history to update it by inserting commits

- `git pull --rebase`

Reorganize the history to update it by inserting commits

- `git rebase -i <SHA1>`

Rewrite the history: delete commits, rewrite commit messages, merge commits

`reword` : rewrite commit message

`drop` : delete a commit

`squash` : merge commits and rewrite message

`fixup` : merge commits but keep old message

- `file(s) edition`
`git add <files...>`
`git rebase --continue`

Manage a conflict during a rebase

Other commands

- `git cherry-pick <SHA1>...`

Replay one or more commits on the current HEAD

- `git add -p <file>...`

Add block-by-block changes to cache

- `git clean`

Delete all new files

- `git cat-file -p <SHA1>`

Display the content of an object

- `git grep <string> -- <path>`

Search for a string in the files contained in the working area

- `git reflog`

View HEAD movement history

Manage tags

- `git tag`

List tags

- `git tag [-a] <tag> <SHA1>`

Create a tag

- `git tag -d <tag>`

Delete a local tag

- `git push --delete origin <tag>`

Delete a remote tag

- `git push origin <tag>`

Share tags

Git Cheat sheet

hupstream campus

Manage sub-modules

- `git submodule add <url> <directory>`

Add a submodule

- `git clone --recursive <url>`

Clone a repository with submodules

- `git fetch`
  `git log --oneline origin/<branch>`
  `git checkout <SHA1 | tag>`

Update submodule content

```
git commit
git push
```

Commit new submodule status in main project

- `git submodule deinit <submodule>`
  `git submodule update --init`

Desactivate / Reactivate submodules

- `git submodule status`

Submodules status

- `git submodule summary`

List all modifications on submodules

- `git checkout --recurse-submodules`

Change branch and update submodules

- `git fetch --recurse-submodules`

Fetch both main repository and all submodules

- `git submodule foreach`

Execute a shell command in all submodules

LFS (Large File Storage)

- `git lfs install`

Add LFS hooks

- `git lfs track <pattern>`

Add file pattern to be tracked with LFS

- `git lfs untrack <pattern>`

Remove pattern to be tracked with LFS

- `git lfs fetch --recent`

Download more LFS objects

git config parameters for submodules

- `status.submoduleSummary true`

Improve git status to add submodules information

- `diff.submodule log`

Improve git diff to take into account submodules

- `submodule.recurse true`

Use recursive update for fetch and checkout commands

Reuse conflict resolution

- `git config --global rerere.enabled true`

Enable rerere

Notes

- `git notes add <SHA1>`

Add a note to a commit

- `git notes append <SHA1>`

Amend an existing note

- `git notes --ref <category> add <SHA1>`

Add a note using a given category

- `git log --oneline --notes=<category>`

git history including notes of a category

- `git push origin refs/notes/*`

Push all notes of any category

Filter-repo

- `git filter-repo --invert-paths \`
  `--path <file, directory> ...`

Delete files, directories

- `git filter-repo \`
  `--subdirectory-filter <directory>`

Split a git repository

- `git filter-repo \`
  `--strip-blobs-bigger-than <size>`

Delete files whose size is bigger than <size>