Cheatography

Base Git Cheat Sheet

by Michele Sacchetti (mikesac) via cheatography.com/34000/cs/10643/

Global Configuration

git config --global user.name "John Doe"	Globally configure your committer name
git config --global user.email johndoe@example.com	Globally configure your committer email

Execute these step **before** any other action with your git client

Initialize new repository

git init	Initialize current directory as a new git repostiroy
git remote add origin https://repourl/repo.git	Add remote repository with name 'origin'
git add --all	Promote all folder content for commit
git commit -am"initial commit"	Commit all files with given comment

Clone existing remote repostiroy

git clone https://giturl/repo.git	Clone remote repository via HTTPS, access credential could be reqeuired
git clone ssh://giturl/repo.git	Clone remote repository via SSH, private/public key exchange is required

Inspect your work

git status	Provide you with the difference in commits and push between your local repo and the remote origin
git log	List latest commit on the local repository
git diff ${FILENAME}	Provide a diff view between the local file and the HEAD version on the remote origin

Save you work

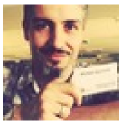
git pull origin develop	Retrieve all commit from remote repository named "origin" and branch "develop" and merge them into local repository
git add --all	Promote ALL unstaged files for commit - **use with care!**
git add ${FILENAME}	Promote given file/folder for commit
git commit -m"my details on commit"	Commit all promoted file to **local** repository
git push origin develop	Push all commit to remote repository named "origin" and branch "develop". Branch will be created if not present. May fail if remote repository has not been pulled before pushing since local repository is not up-to-date

Branching

git branch -a	List all avaiable branches
git fetch	Fetch all remote repository data into local one, including new branches
git fetch -p	Fetch all remote repository data into local one, including new branches.**-p** (prune) causes unexisting remote branches to be dropped also locally
git checkout -b newbranch	create a new branch named "newbranch" starting from current commit
git merge otherbranch	Merge **local** branch named "otherbranch" to be merged into current local branch
git push origin newbranch	Push all commit in **local** branch newbranch to remote origin, creating **remote** branch if not already present
git branch -D oldbranch	Delete **local** branch named "oldbranch". Any tag pushed from deleted branch will be preserved.
git push origin --delete oldbranch	Delete **remote** branch named "oldbranch". Any tag pushed from deleted branch will be preserved.
git merge --squash featurebranch	Merge branch named "featurebranch" into current one, squasshing all commit into a single one. Commit with comment is needed to complete operation.

Tagging

git tag -l	List all avaiable tags
git fetch --tags	Fetch all remote repository data into local one, including tags
git fetch --tags -p	Fetch all remote repository data into local one, including new tags.**-p** (prune) causes unexisting remote tags to be dropped also locally
git tag mytag	Creates a new tag named "mytag" and attach it to current **local** commit
git push origin --tags	Push **local** tags to remote origin
git tag -d oldtag	Delete **local** tag named "oldtag".
git push origin :refs/tags/oldtag	Delete **remote** tag named "oldtag".



Cleaning up your repository

git reset --soft origin master	Reset local repository commit to remote origin on branch master. All changes will be preserved as uncommitted files
git reset --hard origin master	Reset local repository commit to remote origin on branch master. All changes will be discarted -**use with care!**
git clean --dry-run	Remove untracked files - show only candidates no actual removal (alias **-n**)
git clean -f	Remove untracked files - Actual remove **use with care!**
git clean -f -d	Remove untracked files and folders - Actual remove **use with care!**
git clean -f -X	Remove ignored files - Actual remove **use with care!**
git clean -f -x	Remove ignored and non-ignored files - Actual remove **use with care!**
git log -- ${FILENAME}	Show history for ${FILENAME} even if deleted, useful to know when a file has been removed