Cheatography

Git Commands Cheat Sheet

by sertalpbilal via cheatography.com/121942/cs/22478/

Main Commands

git init	Create git repository here (decentralized)
git init --bare	Create bare git repository (centralized)
git clone *repo*	Clone repository at address *repo*
git clone -b *branch repo*	Clone *branch* from repository

Status

git status	List changes
git status --short --branch	List changes (compact)
git log	Display commit history
git log --oneline --decorate --graph --all	Display visual commit history
git config --global core.autocrlf true	Ignore OS-specific line endings

Commits

git add *hello.py*	Add *hello.py* (or stage changes) for next commit
git rm --cached *hello.py*	Remove *hello.py* from git tracking
git checkout -- *hello.py*	Discard all local changes to *hello.py*
git add -u	Stage all changes under tracking
git add -A	Stage all changes (new files and deletions included)
git update-index --assume--unchanged *hello.py*	Ignore changes to *hello.py* locally
git add --interactive	Review and stage changes
git commit	Create a commit (snapshot)
git commit -m "*message*"	Create a commit with message
git commit -a	Commit every change under tracking
git commit -ammend	Combine current and last commit (correction)
git diff	Show all unstaged differences
git diff --cached	Show all differences including staged ones

Commits (cont)

git diff *branch1..branch2*	See differences between *branch1* and *branch2*

You can use commits instead of branch names. *HEAD* refers to current point in git.

Reseting

git reset --soft *commit*	Stage all changes since *commit* (used for squashing)
git reset --hard	Discard all local changes and reset to last commit

Maintenance

git revert *commit*	Creates a new commit out of an old commit (turn back in history)
git clean -dfx	Delete all untracked files and directories ! Dangerous
git clean -dfxn	Dry run - Show which files will be deleted
git gc --aggressive --prune=now	Reduce file size of repository by pruning
git checkout *file*	Discard local changes to *file*
bfg -delete-files *file*	BFG cleans sensitive data *file*

Remote Management

git remote add *target repo*	Add remote *target* at address *repo*
git remote -v	List all remotes
git remote set-url *target repo*	Changes the address of remote *target*
git push -f *origin commit:master*	Forces *origin/master* branch to be moved to *commit*
git pull (--rebase) *target branch*	Push existing commit to *target/branch*
git push *target branch*	Push *branch* to *target/branch*
git push *target branch1:branch2*	Push local *branch1* to *target/branch2*
git push -u *target branch*	Push and track *target/branch*. It sets upstream for default pull / push
git push *target* --delete *branch*	Deletes *target/branch* at remote

Cheatography

Git Commands Cheat Sheet

by sertalpbilal via cheatography.com/121942/cs/22478/

Remote Management (cont)

git fetch *target*	Update branch information of *target* remote

Note: *repo* might start with `git@` or `https`. When it starts with `git@` an SSH key is needed for pull/push operations.

Branch management

git checkout *target*	Switch to *target* branch
git checkout -b *feature*	Create new branch from current commit
git checkout -b *feature master*	Create new branch *feature* from *master* branch
git branch -a	List all branches
git branch -r --merged master	List remote branches that are merged with *master*
git merge *feature*	Merge *feature* branch into current branch
git merge --no-ff *feature* -m "*message*"	Merge *feature* with a commit (no fast-forward)
git branch -d *feature*	Delete *feature* branch (if already merged)
git branch -D *feature*	Force delete *feature* branch ! Dangerous
git branch -f *feature commit*	Move branch head *feature* to *commit* ! Dangerous

Use

```
git for-each-ref --sort=committerdate refs/heads/ --format='%(HEAD) %(refname:short) - %(objectname:short) - %(contents:subject) - %(authorname) (%(committerdate:relative))'
```

to print summary of each branch, last commit, and other info

Tag management

git tag -a *v1.0.0* -m "*message*"	Creates an annotated tag with message
git push *target* --tags	Push local tags to remote *target*
git tag -d *v1.0.0*	Delete local tag *v1.0.0"
git push *target* :refs/tags/*v1.0.0*	Delete remote tag *target/v1.0.0*

Semantic Versioning

git describe --tags --dirty	Describe current commit using tags

Submodules

git submodule update --init --recursive .	Pull all submodules (linked repositories)
git submodule status *folder*	Show the status of submodule under *folder*