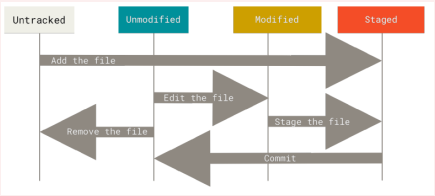
Cheatography

Git Cheat Sheet

by Loreno10 via cheatography.com/124704/cs/23866/

States



Diff

`git diff`	WORKSPACE vs STAGED
`git diff --staged` (or `--cached`)	STAGED vs COMMITED
`git diff HEAD`	WORKSPACE + STAGED vs COMMITED
`git diff <SHA-1>`	WORKSPACE vs any commit (more general case of the above)
`git diff master...topic`	Changes introduced if topic gets merged into master
`-w`	IGNORE whitespace

Removal

`git rm README`	Remove file from git and locally
`git rm --cached README`	Remove file from git, but keep locally as UNTRACKED
`git reset HEAD README`	UNSTAGE a file
`git checkout -- README`	Revert a file back to its form from the last commit

Stashing

`git stash`	Push all TRACKED modified files (`-u` to include untracked, `-a` to include ignored)
`git stash list`	LIST all stashes
`git stash apply`	APPLY the last stash
`git stash apply stash@{2}`	APPLY a chosen stash
`git stash pop`	APPLY stash and DELETE it
`git stash drop`	DELETE stash

Remotes

Remotes (cont)

`git fetch origin`	FETCH all data from origin
`git pull`	FETCH and MERGE
`git push <origin> <master>`	PUSH branch to a remote
`git branch -vv`	Details of local branches and what they track

Tagging

`git tag`	LIST tags
`-l "119.*"`	FILTER tags
`git tag -a v2.0 -m "Version 2.0"`	CREATE annotated tag
`git tag v2.0`	CREATE lightweight tag
`git show v2.0`	GET tag
`git tag -d v2.0`	DELETE tag
`git push <origin> <tag>`	PUSH single tag
`git push <origin> --tags`	PUSH all tags

Rebasing

`git rebase master`	Takes commits from current branch and applies them on top of master. Master stays on previous commit, only current branch is moved

Git Log

`git remote -v`	LIST remotes with URLs
`git remote show <or igi n>`	GET details with tracked branches
`git remote add <or igi n> <ur l>`	ADD a remote
`git remote rename <or igi n> <up str eam >`	RENAME remote origin to upstream
`git remote rm <or igi n>`	DELETE remote
`git push <or igi n> --delete <fe atu re/ one>`	DELETE branch from remote

`-p`

`-2`

`master featur e/one`

`--all`

`--stat`

`--pret ty= one lin e/s hor t/f ull /fu lle r/f orm at {some format}`

`--oneline`

`--graph`

`--sinc e=2.we eks /20 10- 01-15` (or `after`)

`--unti l=2021` (or `before`)

`--auth or=Joe`

`--grep =word`

`-S word`

Example: `git log --pretty="%h - %s" --author='Junio C Hamano' --since="2008-10-01" \
--before="2008-11-01" --no-merges`

By **Loreno10**
cheatography.com/loreno10/

Published 7th December, 2020.
Last updated 20th December, 2020.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

–pretty=format options

Option	Description of Output
%H	Commit hash
%h	Abbreviated commit hash
%T	Tree hash
%t	Abbreviated tree hash
%P	Parent hashes
%p	Abbreviated parent hashes
%an	Author name
%ae	Author email
%ad	Author date (format respects the –date=option)
%ar	Author date, relative
%cn	Committer name
%ce	Committer email
%cd	Committer date
%cr	Committer date, relative
%s	Subject

Configuration

Git configuration is stored in 3 places:

- `/etc/g itc onfig` (`--system`)
- `~/.git config` (`--global`)
- `.git/c onfig` (`--local`)

Each level overwrites the previous one.

Setting up:

```
git config --global user.name "John Doe"
git config --global user.email johndo e@e xam ple.c
om
git config --global core.e ditor code
```

Check some setting:

```
git config user.name
```

See all configuration and sources:

```
git config --list --show -origin
```


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com