

git cheat sheet

getting started

start a new repo:
`git init`
clone an existing repo:
`git clone $URL`

know where you are

`git status`

prepare to commit

add untracked file:
(or unstaged changes)
`git add $FILE`
add ALL untracked files and unstaged changes:
`git add .`
choose which parts of a file to stage:
`git add -p`

delete or move file:
`git rm $FILE`
`git mv $OLD $NEW`
tell git to forget about a file without deleting it:
`git rm --cached $FILE`
unstage everything:
`git reset HEAD`

make commits

make a commit:
(and open a text editor to write the message)
`git commit`
make a commit:
`git commit -m 'message'`
commit all unstaged changes:
`git commit -am 'message'`

git has 17 million options but this is how I use it!

move between branches

switch branches:
`git switch $NAME OR`
`git checkout $NAME`
create a branch:
`git switch -c $NAME OR`
`git checkout -b $NAME`

list branches:
`git branch`
delete a branch
`git branch -d $NAME`
force delete a branch:
`git branch -D $NAME`
list branches by most recently committed to:
`git branch --sort=committerdate`

look at a branch's history

log the branch
`git log main`
show how two branches relate to each other:
`git log --graph a b`
one line log:
`git log --oneline`

code archaeology

show who last changed each line of a file:
`git blame $FILENAME`
show every commit that modified a file:
`git log $FILENAME`
find every commit that added or removed some text:
`git log -S banana`

diff commits

show diff between a commit and its parent:
`git show $COMMIT_ID`
show diff between a merge commit and its merged parents:
`git show --remerge-diff $COMMIT_ID`
diff two commits:
`git diff $COMMIT_ID $COMMIT_ID`
just show diff for one file:
`git diff $COMMIT_ID $FILENAME`
show a summary of a diff:
`git diff $COMMIT_ID --stat`
`git show $COMMIT_ID --stat`

diff staged/unstaged changes

diff all staged and unstaged changes:
`git diff HEAD`
diff just staged changes:
`git diff --staged`
diff just unstaged changes:
`git diff`

configure git

set a config option:
`git config user.name 'Julia'`
name ↑ value ↑
see all possible config options:
`man git-config`
set option globally:
`git config --global ...`
add an alias:
`git config alias.st status`

important git files

local git config:
`.git/config`
global git config:
`~/.gitconfig`
list of files to ignore:
`.gitignore`

trash your changes

delete all staged and unstaged changes to one file:
`git checkout HEAD $FILE`
delete unstaged changes to one file:
`git checkout $FILE`
delete all staged and unstaged changes:
`git reset --hard`
delete untracked files:
`git clean`
"stash" all staged and unstaged changes (pretend I might get them back later)
`git stash`

edit history

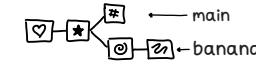
"undo" the most recent commit (keep your working directory the same):
`git reset HEAD^`
squash the last 5 commits into one:
`git rebase -i HEAD^^^^^`
(and change "pick" to "fixup" for any commit I want to combine with the previous one)
undo a failed rebase:
① `git reflog BRANCHNAME`
② do a painstaking search
③ `git reset --hard $COMMIT_ID`
change a commit message:
(or add a file you forgot)
`git commit --amend`

restore an old file

get the version of a file from another branch or commit
`git checkout $COMMIT_ID $FILE`
OR
`git restore $FILE --source $COMMIT_ID`

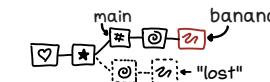
combine diverged branches

how the branches look before:



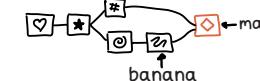
→ combine with rebase:

`git switch banana`
`git rebase main`



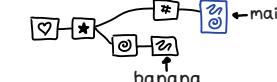
→ combine with merge:

`git switch main`
`git merge banana`
`git commit`



→ combine with squash merge:

`git switch main`
`git merge --squash banana`
`git commit`



add a remote

`git remote add $NAME $URL`

push your changes

push the main branch to the remote origin:
`git push origin main`

push a branch to the remote origin that you've never pushed before:
`git push -u origin $NAME`

push the current branch to its remote "tracking branch":
`git push`

force push:
`git push --force-with-lease`
push tags:
`git push --tags`

pull changes

fetch changes:
(but don't change any of your local branches)
`git fetch origin main`

fetch changes and then merge them into your current branch:
`git pull origin main OR`
`git pull`

fetch changes and then rebase your current branch:
`git pull --rebase`

fetch all branches:
`git fetch --all`

ways to refer to a commit

every time we say `$COMMIT_ID`, you can use any of these:

★ a branch	main
★ a tag	v0.1
★ a commit ID	3e887ab
★ a remote branch	origin/main
★ current commit	HEAD
★ 3 commits ago	HEAD^^^
★ 3 commits ago	HEAD~3

