git --print-out

Git Cheat Sheet

Getting Started

Start a new repo:
```
git init
```
Clone an existing repo:
```
git clone <url>
```

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
```
Move file:
```
git mv <old> <new>
```
Delete file:
```
git rm <file>
```
Tell Git to forget about a file without deleting it:
```
git rm --cached <file>
```
Unstage one file:
```
git reset <file>
```
OR
```
git restore --staged <file>
```
Unstage everything:
```
git reset
```
OR
```
git restore --staged .
```
Check what you added:
```
git status
```

Make Commits

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

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
```
List branches by most recently committed to:
```
git branch --sort=-committerdate
```
Delete a branch:
```
git branch -d <name>
```
Force delete a branch:
```
git branch -D <name>
```

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
```

Diff Commits

Show diff between a commit and its parent:
```
git show <commit>
```
Diff two commits:
```
git diff <commit> <commit>
```
Diff one file since a commit:
```
git diff <commit> <file>
```
Show a summary of a diff:
```
git diff <commit> --stat
```
```
git show <commit> --stat
```

Ways to refer to a commit

Every time we say `<commit>`, 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^^^` or `HEAD~3`

Discard Your Changes

Delete unstaged changes to one file:
```
git restore <file>
```
OR
```
git checkout <file>
```
Delete all staged and unstaged changes to one file:
```
git restore --staged --worktree <file>
```
OR
```
git checkout HEAD <file>
```
Delete all staged and unstaged changes:
```
git reset --hard
```
OR
```
git restore --staged --worktree .
```
Delete untracked files:
```
git clean
```
'Stash' all staged and unstaged changes:
```
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~5
```
Then change "pick" to "fixup" for any commit you want to combine with the previous one

Undo a failed rebase:
```
git reflog BRANCHNAME
```
Then manually find the right commit ID in the reflog, then run:
```
git reset --hard <commit>
```
Change a commit message (or add a file you forgot):
```
git commit --amend
```

Code Archaeology

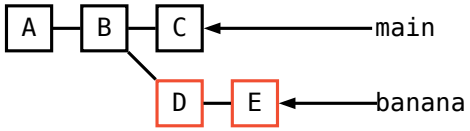
Look at a branch's history:
```
git log main
```
```
git log --graph main
```
```
git log --oneline
```
Show every commit that modified a file:
```
git log <file>
```
Show every commit that modified a file, including before it was renamed:
```
git log --follow <file>
```
Find every commit that added or removed some text:
```
git log -G banana
```
Show who last changed each line of a file:
```
git blame <file>
```

Combine Diverged Branches

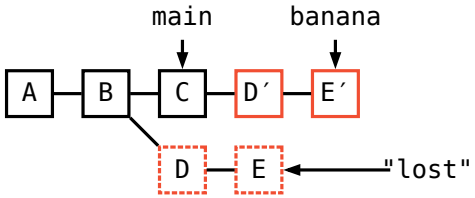
Combine with rebase:

```
git switch banana
git rebase main
```

Before:



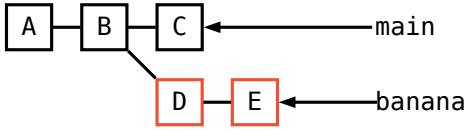
After:



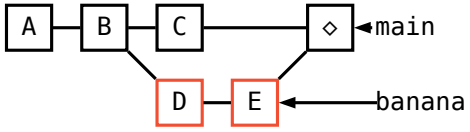
Combine with merge:

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

Before:



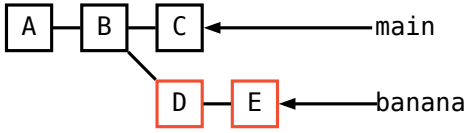
After:



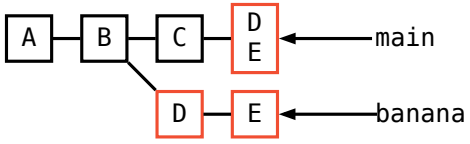
Combine with squash merge:

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

Before:



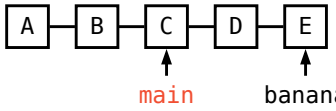
After:



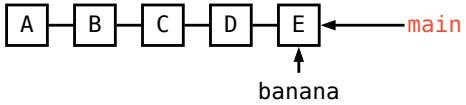
Bring a branch up to date with another branch (aka "fast-forward merge"):

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

Before:



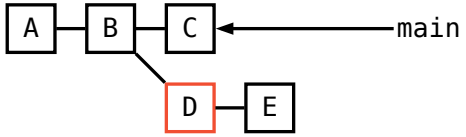
After:



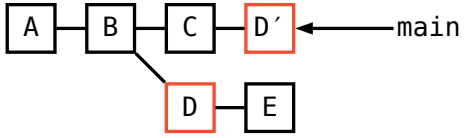
Copy one commit onto the current branch:

```
git cherry-pick <commit>
```

Before:



After:



Restore an Old File

Get the version of a file from another commit:

```
git checkout <commit> <file>
```

OR

```
git restore <file> --source <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 the current branch to its remote "tracking branch":

```
git push
```

Push a branch that you've never pushed before:

```
git push -u origin <name>
```

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 rebase your current branch:

```
git pull --rebase
```

Fetch changes and then merge them into your current branch:

```
git pull origin main
```

OR

```
git pull
```

Configure Git

Set a config option:

```
git config user.name 'Your Name'
```

Set option globally:

```
git config --global ...
```

Add an alias:

```
git config alias.st status
```

See all possible config options:

```
man git-config
```

Important Files

Local git config:

```
.git/config
```

Global git config:

```
~/.gitconfig
```

List of files to ignore:

```
.gitignore
```