Git Cheat Sheet

swipe

Setup

Set the name and email that will be attached to your commits and tags.

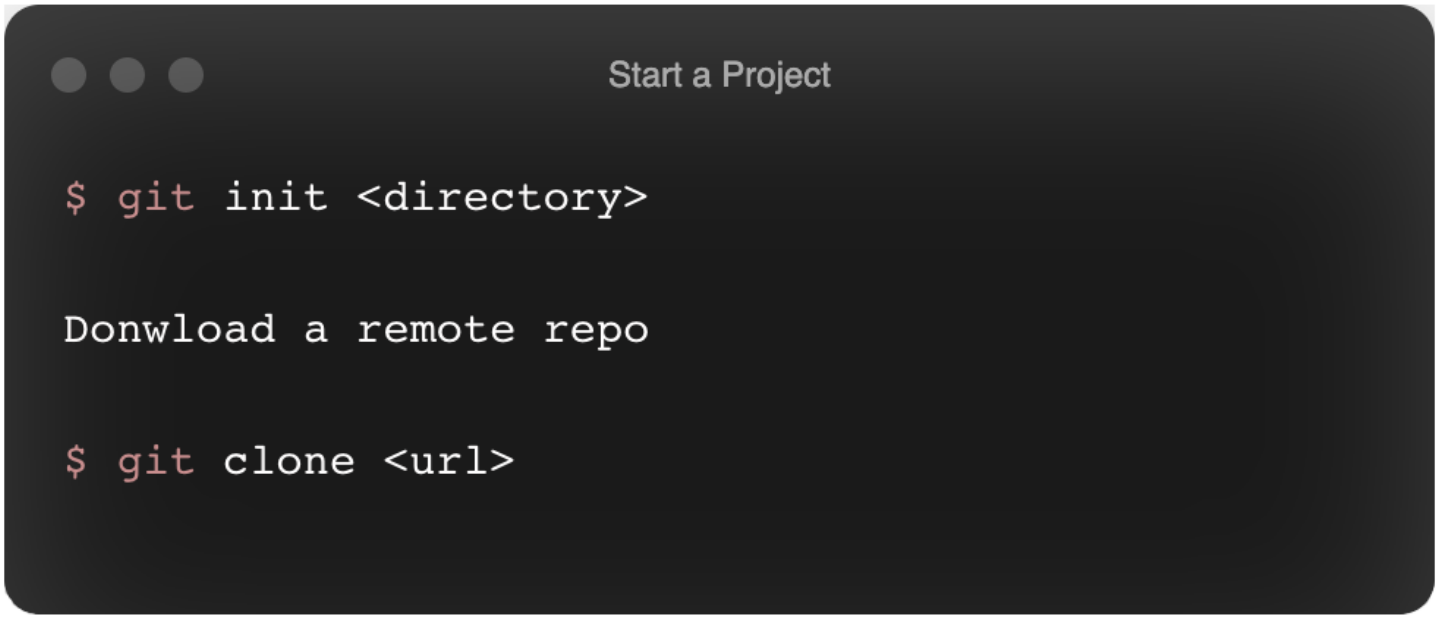
The Setup

```
$ git config --global user.name "The Tech Lead"
$ git config --global user.email "techlead@schbang.com"
```

swipe

Start a Project

Create a local repo (omit <directory>) to initialise the current directory as a git repo



swipe

Make a Change

Make a Change

```
Add a file to staging
$ git add <file>

Stage all files
$ git add .

Commit all stages files to git
$ git commit -m "Commit Message"

Add all changes made to tracked files & commit
$ git commit -am "Commit Message"
```

swipe

Basic Concepts

main : default development branch

origin : default upstream repo

HEAD : current branch

HEAD^ : parent of HEAD

HEAD~4 : great great grandparent of HEAD

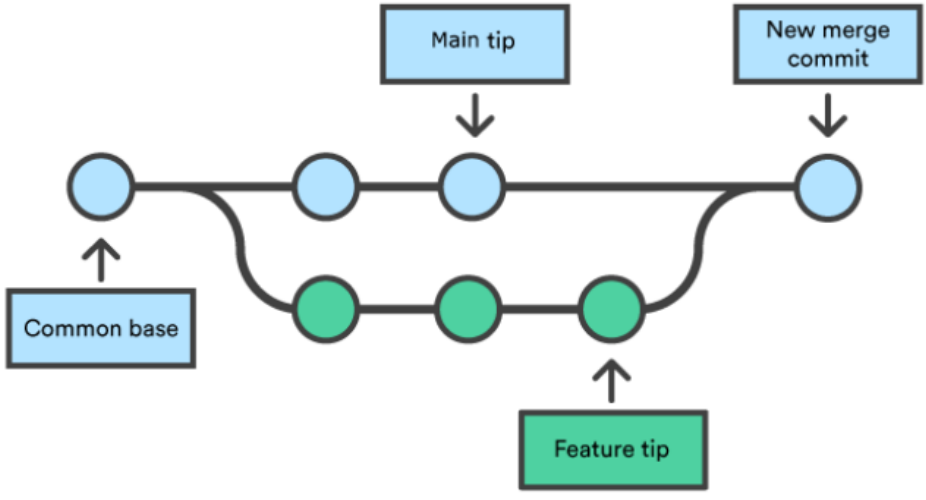
Branches

Branches

```
List all local branches. Add -r flag to show all remote branches,
-a flag to show all branches.
$ git branch

Create a new branch
$ git branch <new-branch>

Switch to a branch & update the working directory
$ git checkout <branch>

Create a new branch and switch to it
$ git checkout -b <new-branch>

Delete a merged branch
$ git branch -d <brach>

Delete a branch, whether merged or not
$ git branch -D <branch>

Add a tag to current commit(often used for new version releases)
$ git tag <tag-name>
```

Merging

Merge branch **a** into branch **b**. Add --no-ff option for no-fast-forward merge



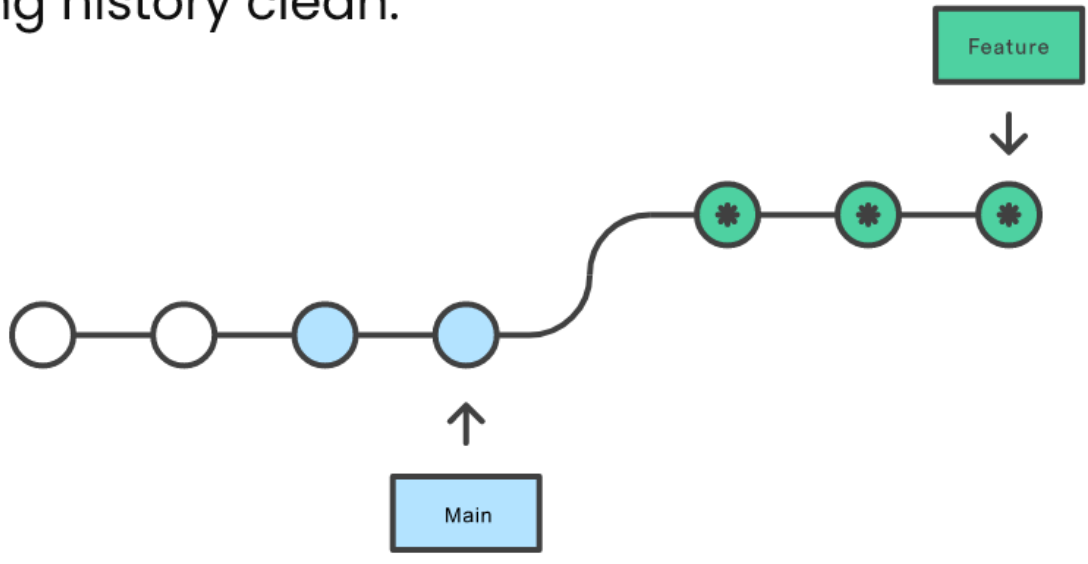
Merging

```
$ git checkout b
$ git merge a

Merge and Squasg all commits into one new commit
$ git merge --squash a
```

swipe

Rebasing

Rebase feature branch onto main (to incorporate new changes made to main). Prevents unnecessary merge commits into feature, keeping history clean.



Rebasing

```
$ git checkout feature
$ git rebase main

Interatively clean up a branches commits before rebasing onto
main
$ git rebase -i main

Interatively rebase the last 3 commits on current branch
$ git rebase -i Head~3
```

swipe

Undoing Things

Undoing Things

```
Move or rename a file & stage move
$ git mv <existing_path> <new_path>

Remove a file from working directory & staging area, then stage the
removal
$ git rm <file>

Remove from staging area only
$ git rm --cached <file>

View a previous commit (READ only)
$ git checkout <commit_ID>

Create a new commit, reverting the changes from a specified commit
$ git revert <commit_ID>

Go back to a previous commit & delete all commits agead of
it(revert is safer). Add --hard flag to alsp delete workspace
changes (BE VERY CAREFUL)

$ git reset <commit_ID>
```

Review your Repo

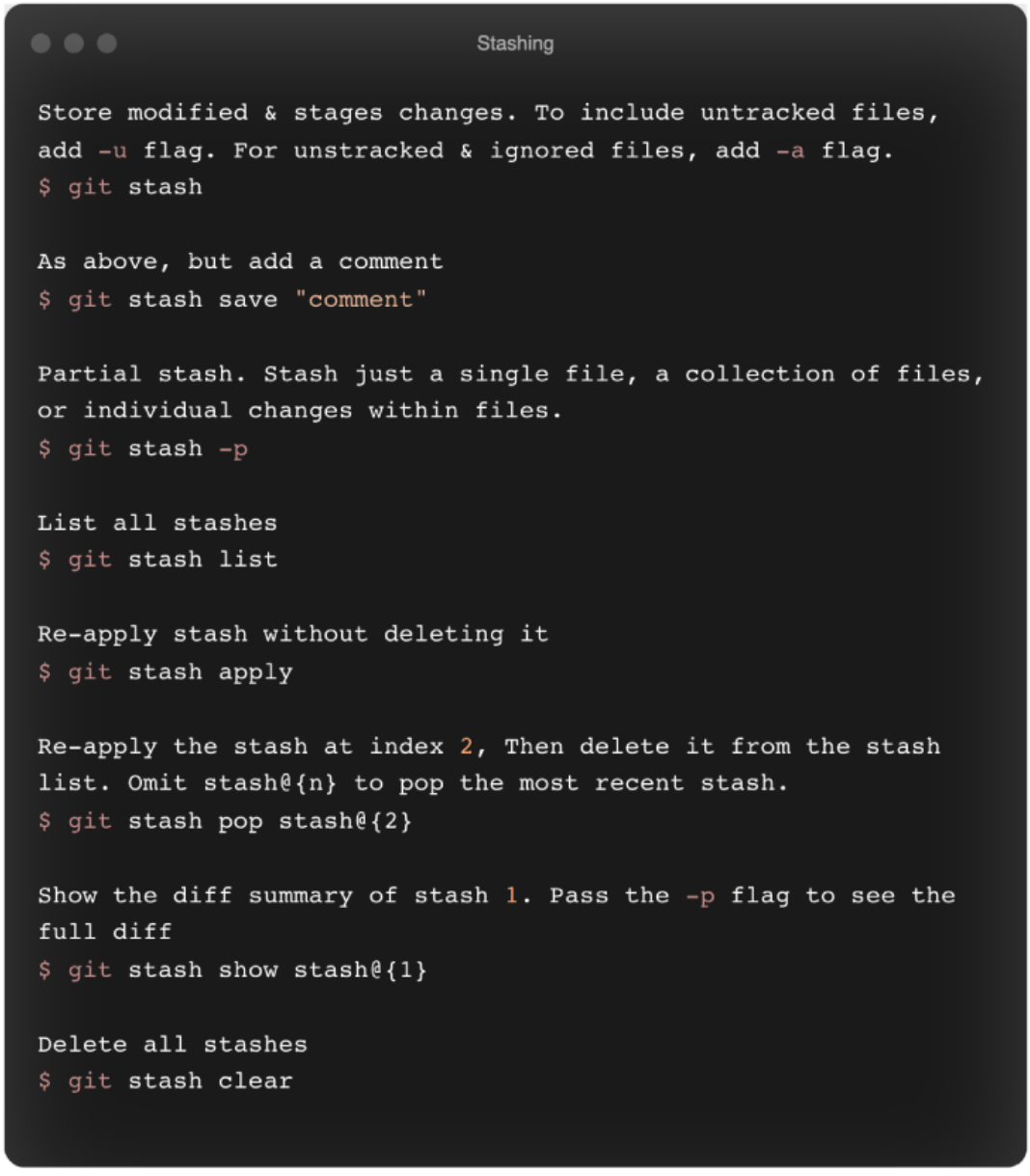
Repo Review

```
List new or modified files not yet committed
$ git status

List commit history with respective IDs
$ git log --oneline

Show changes to unstaged files, for changes to stages files, add
--cached option
$ git diff

Show changes between two commits
$ git diff commit1_ID commit2_ID
```

swipe

Stashing

git stash temporarily shelves (or stashes) changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on.



swipe

Synchronizing

Synchronizing

```
Add a remote repo
$ git remote add <alias> <url>

View all remote connections. Add -v flag to view urls.
$ git remote

Remove a connection
$ git remote remove <alias>

Rename a connection
$ git remote rename <old> <new>

Fetch all branches from remote repo (no merge)
$ git fetch <alias>

Fetch a specific branch
$ git fetch <alias> <branch>

Fetch the remote repo copy of the current branch, Then merge
$ git pull

Merge your local changes onto the top of new changes made to
the remote repo.
$ git pull --rebase <alias>

Upload local content to remote repo
$ git push <alias>

Upload to a branch
$ git push <alias> <branch>
```

@the.tech.lead

Follow for more tips like this

Let us know what you think in the comments!

Give us some love

Save for later