GIT CHEATSHEET

① YOU ALREADY KNOW ABOUT:

```
$ git init
$ git add -A
$ git commit -m "my commit"
$ git status
$ git log
```

YOU CAN ALSO DO:

```
$ git remote add origin https://github.com/user/myrepo.git
$ git push -u origin master
$ git push
```

IF YOU WANT A LOCAL VERSION OF A REMOTE REPO:

```
$ git clone https://github.com/user/myrepo.git
```

> IT IS IMPORTANT TO NOTE THAT YOU GET NO CREDIT (THE GREEN SQUARES ON YOUR GITHUB PROFILE) FOR FORKED PROJECTS. YOU SHOULD CLONE THE REPO, IF YOU WANT YOUR ACTIVITY TO SHOW.

② DIFFERENCES BETWEEN COMMITS

- DIFFERENCE BETWEEN CURRENT STATE OF FILES AND LAST COMMIT:

```
$ git diff
```

- DIFF BETWEEN CURRENT STATE AND COMMIT BEFORE LAST:

```
$ git diff HEAD~1
```

3 CHANGING A FILE TO WHAT IT WAS LIKE AT THE TIME OF THE LAST COMMIT

IN CASE WE MESSED UP BADLY

- RESETTING A SINGLE FILE:

```
$ git checkout README.txt
```

- RESETTING WHOLE PROJECT (CAREFUL!):

```
$ git reset --hard
```

4 I DON'T LIKE MY LAST COMMIT AND WANT TO REDO IT WITHOUT LOSING CHANGES

- GOING BACK TO WHEN WE WERE ABOUT TO COMMIT:

```
$ git reset --soft HEAD~1
```

5 GOING BACK TO AN OLDER COMMIT AND DISCARDING CHANGES

OR WHATEVER

- `$ git reset --hard HEAD~3`

6 WORKING WITH .gitignore

- IGNORING SOMETHING ANYWHERE IN THE PROJECT:

```
something/
```

(← INSIDE .gitignore)

- IGNORING SOMETHING RELATIVE TO THE ROOT OF THE APP:

```
/something
```

7 SHOW COMMITS LOG IN A CONDENSED WAY

- `$ git log --oneline`

8 VISUALIZING PROJECT BRANCHES GRAPHICALLY

- `$ git log --graph --oneline --all --decorate`
- TOO MUCH TYPING? CREATE AN ALIAS!

  `$ git config --global alias.whatever "log --graph ETC..."`

9 FUN STATS

- SEE NUMBER OF COMMITS:

  `$ git log --oneline | wc -l`
- FIND THE SLACKERS IN YOUR GROUP!

  COMMITS BY AUTHOR:

  ```
  $ git shortlog
  $ git shortlog -sne
  ```

10 INSPECTING COMMITS

- INFO ABOUT LAST COMMIT:

  `$ git show HEAD`
- INFO ABOUT AN OLDER COMMIT:

  ```
  $ git show HEAD~10   OR WHATEVER
                       OR HASH
  ```

11 WHAT ARE THE REMOTES FOR MY REPO?

- `$ git remote`
- `$ git remote -v`