Introduction of Git

汪杰

Contents

Git ~~Cheat Sheet~~ Check List

```
git clone https://github.com/link/to/your/repo
# Enter your UserName and PassWord (token) if the repo is private
cd repo # change "repo" to your repository name

# make some changes now
git add .
git commit -m "your commit message"

# make more changes
git add .
git commit -m "more commit messages"

# a lot of changes and commits...

git push # hand in! DO THIS BEFORE DEADLINE!
# Enter your UserName and PassWord (token)
```

See `man giteveryday` for more.

Prerequisites

- Setup tokens on your github first: Creating a personal access token on github.
- Config your name and email: `vim ~/.gitconfig` . Add the following contents to `.gitconfig`

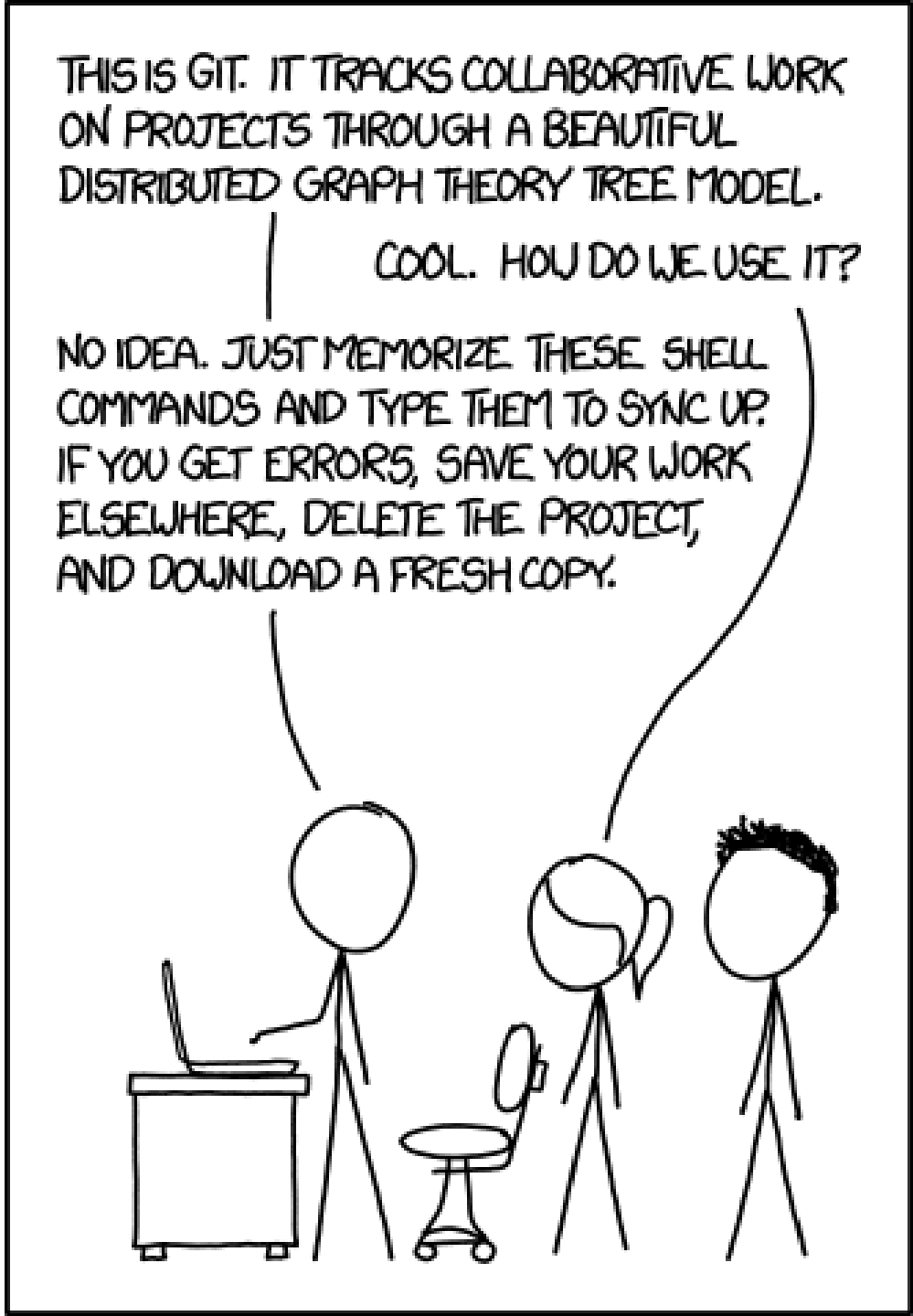
```
[user]
    name = your name
    email = your_email@xxx.xxx
```

Contents

- Git checklist
- Git commands
- VSCode git extensions

What is Git ?

- [distributed] Version Control System (VCS)
- help maintain a history of changes
- also maintain metadata associated with each snapshot, like who created each snapshot, messages, etc.



Basics

- `man git-<command>` : get help for a git command
- `git init` : creates a new git repo, with data stored in the `.git` directory
- `git status` : tells you what's going on
- `git add <filename>` : adds files to staging area
- `git commit` : creates a new commit
  - Write good commit messages!
  - Even more reasons to write good commit messages!
- `git log` : shows a flattened log of history
  - `git log --all --graph --decorate` : visualizes history as a DAG
- `git diff <filename>` : show changes you made relative to the staging area
- `git diff <revision> <filename>` : shows differences in a file between snapshots
- `git checkout <revision>` : updates HEAD and current branch

Branching and merging

- `git branch` : shows branches
- `git branch <name>` : creates a branch
- `git checkout -b <name>` : creates a branch and switches to it same as `git branch <name>; git checkout <name>`
- `git merge <revision>` : merges into current branch
- `git mergetool` : use a fancy tool to help resolve merge conflicts
- `git rebase` : rebase set of patches onto a new base

Remotes

- `git remote` : list remotes
- `git remote add <name> <url>` : add a remote
- `git push <remote> <local branch>:<remote branch>` : send objects to remote, and update remote reference
- `git branch --set-upstream-to=<remote>/<remote branch>` : set up correspondence between local and remote branch
- `git fetch` : retrieve objects/references from a remote
- `git pull` : same as `git fetch; git merge`
- `git clone` : download repository from remote

Undo

- `git commit --amend` : edit a commit's contents/message
- `git reset HEAD <file>` : unstage a file
- `git checkout -- <file>` : discard changes
- See Oh Shit, Git!?! for more ways to undo changes

Advanced Git

- `.gitignore` : specify intentionally untracked files to ignore
- **Workflows**: Some practices to follow when working on big projects (and there are many ----- different ------ approaches).
- Shell integration: it's super handy to have a Git status as part of your shell prompt (zsh, bash). Often included in frameworks like Oh My Zsh.

Contents

- Git checklist
- Git commands
- VSCode git extensions

VSCode git extensions

- Search "git" at extension marketplace
- Try GitLens, Git Graph...

GitLens
Git supercharged

Recommendations

- Learning Git commands interactively
  - https://learngitbranching.js.org/?locale=zh_CN
- The Missing Semester of Your CS Education
  - https://missing.csail.mit.edu/2020/version-control/
  - https://www.bilibili.com/video/BV1164y1u7mP?p=6
- Git Commands Explained with Cats!
  - https://girliemac.com/blog/2017/12/26/git-purr/?continueFlag=093a4862fca21eb13281e72de07c666f
- https://www.liaoxuefeng.com/wiki/896043488029600