VIA University College

DNP-A18

Version Control

How to collaborate on software

Version Control

What is a version control system?

Git

How to share and collaborate on source code using the Git version control system.

GitHub

Learn how to host your Git repositories online

Git & VSCode

How can version control be an integrated part of your daily workflow?

Exercises

Start using version control yourself!

Why is Collaboration Important?

How to Collaborate?

How do YOU collaborate today?

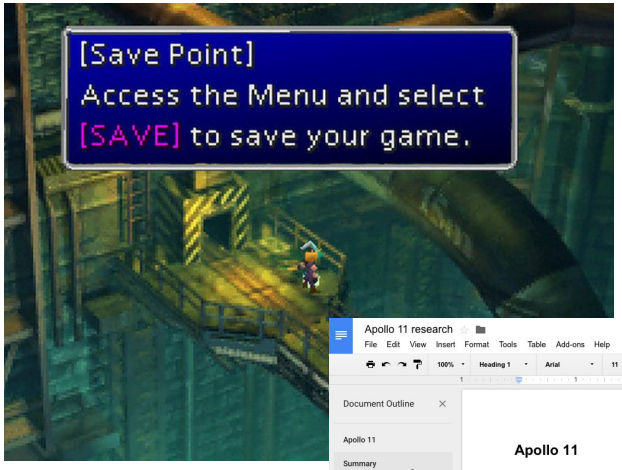
What is Version Control?

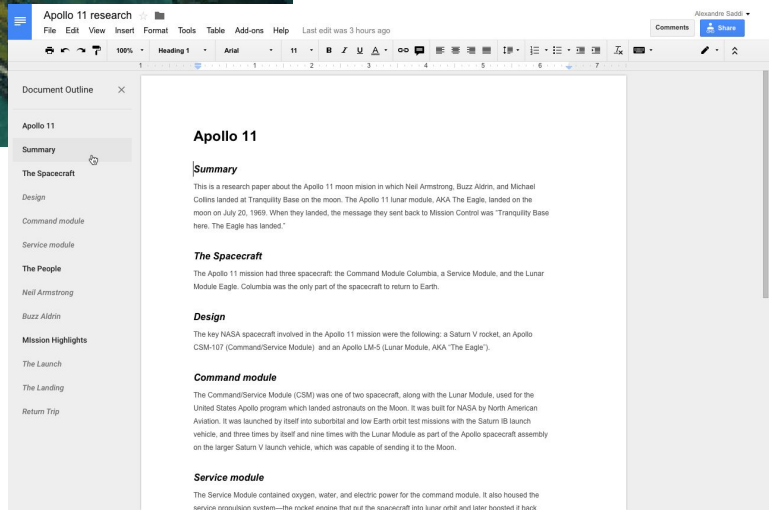
Version control is **confidence** and **freedom**.

Management of different **versions** of content

There are two main types of version control system models:

- The **centralized model** - all users connect to a central, master repository
- The **distributed model** - each user has the entire repository on their computer

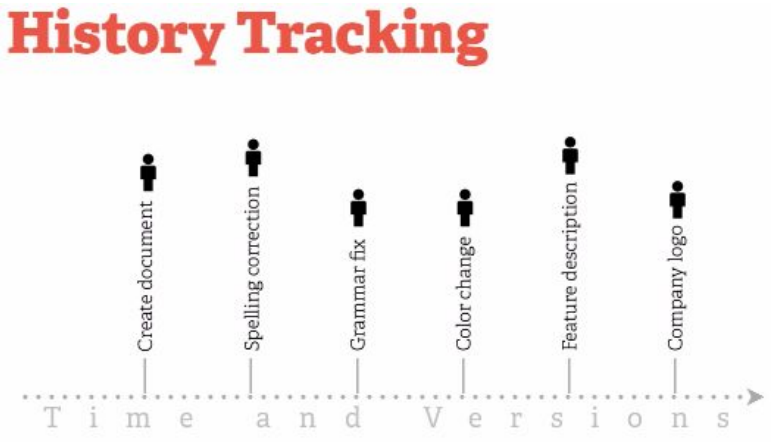


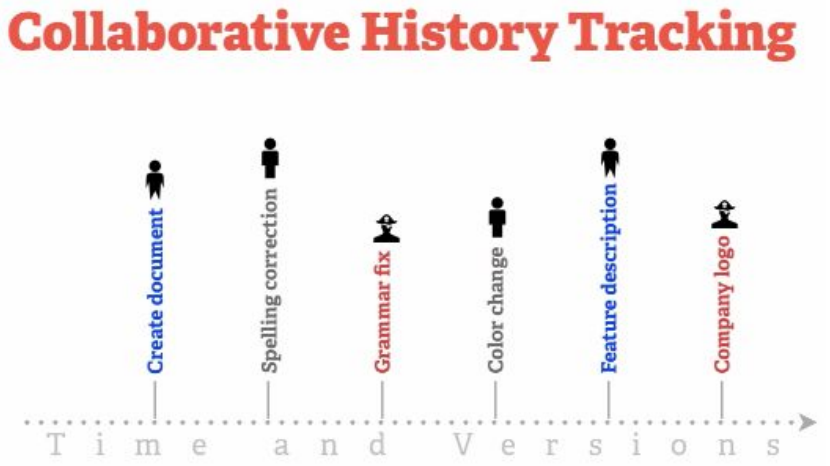


[Version Control Wiki]

What is Version Control?

Saving things, again and again...





Version Control Systems

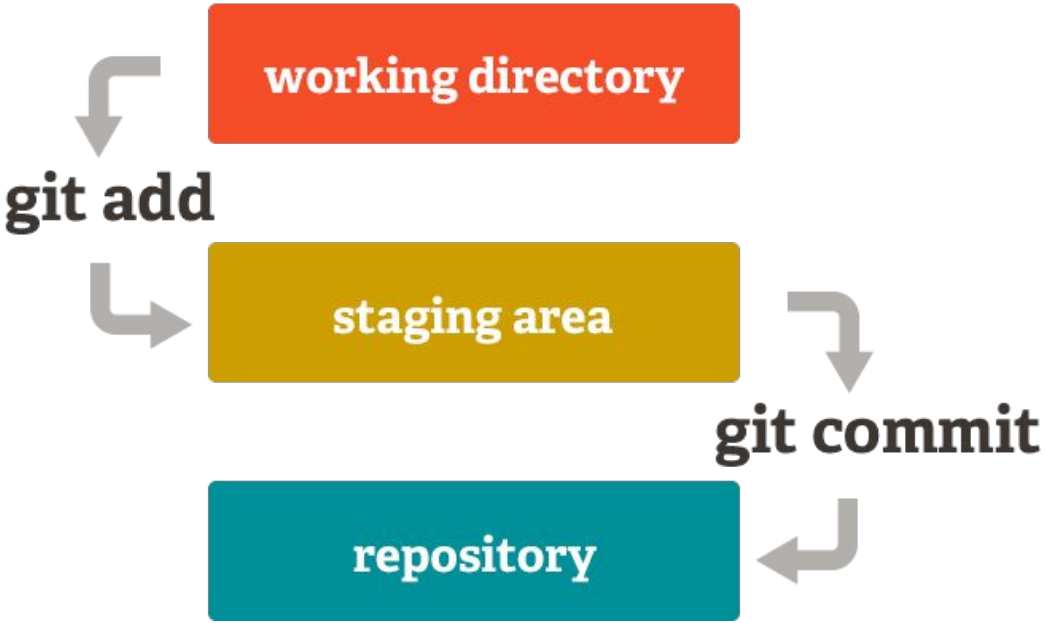
What is Git?

Git is a free and open source version control system

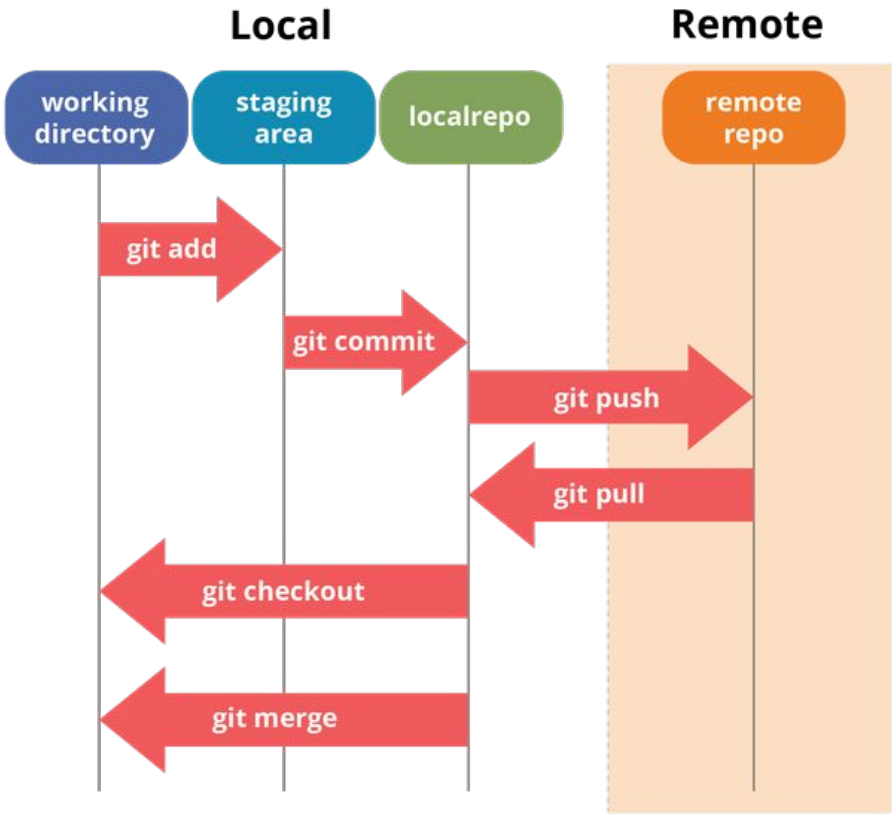
In contrast to traditional SCM's:

- **Content**, not files
- **Opt in**, not opt-out
- **Open**, not locked
- **Distributed**, not centralized
- **Conversations**, not cutoffs
- **People**, not tools
- **Journal**, not backup
- **Anywhere**, not just online

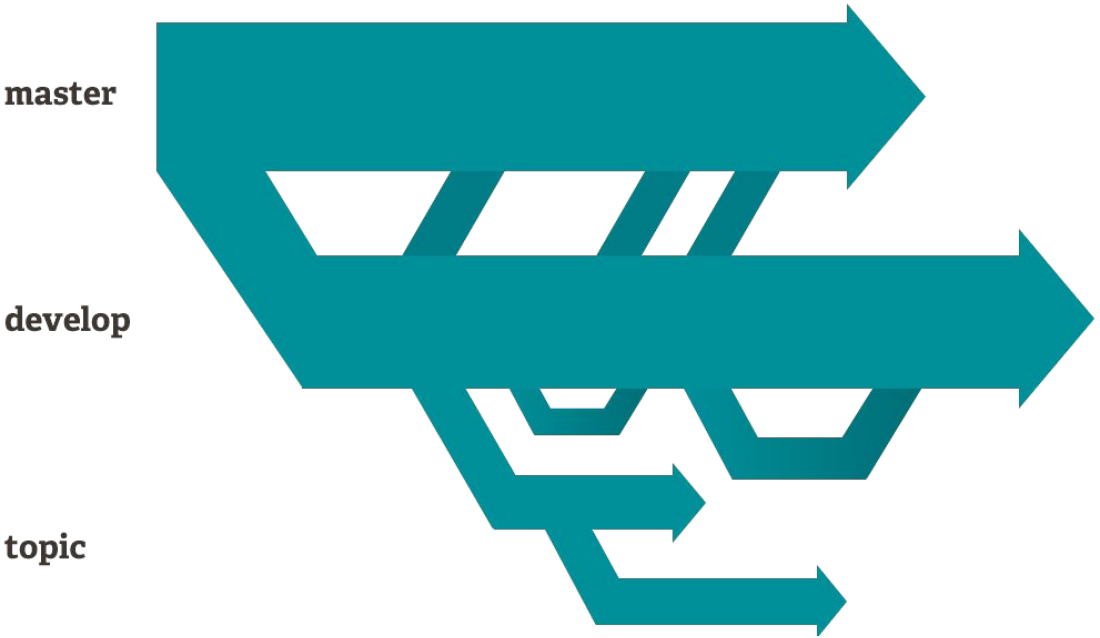
Common Terms



Common Terms



Common Terms



Common Terms

Git

Reference Manual

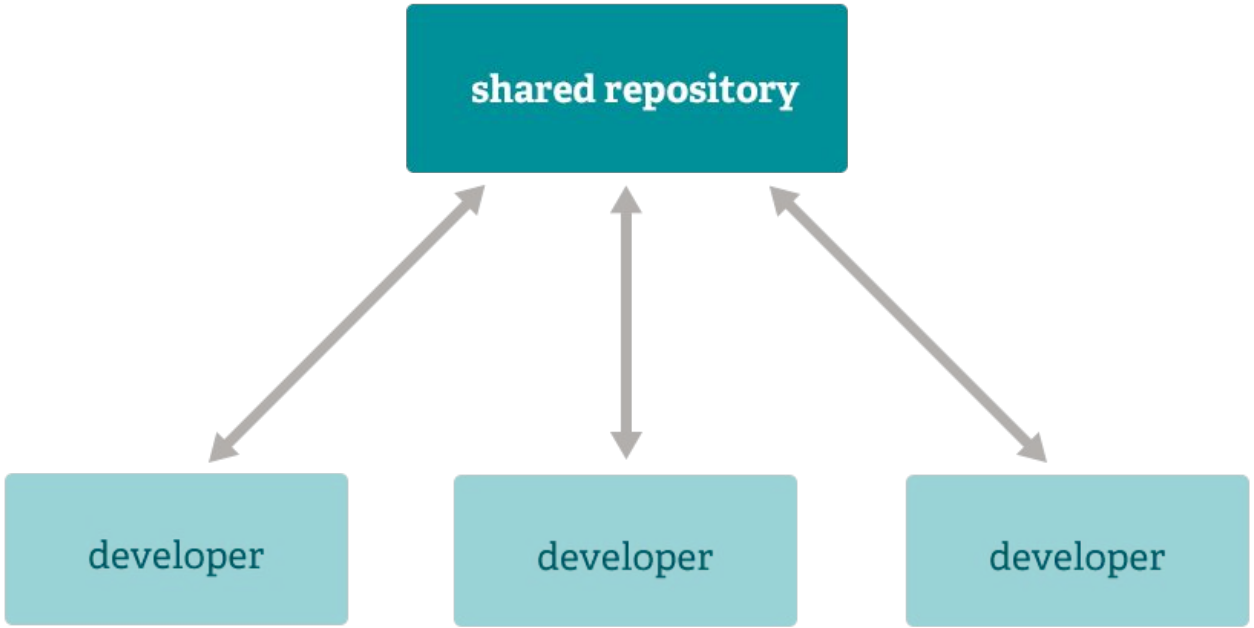
GitHub Cheat Sheet

Visual Cheat Sheet

GitHub Glossary

Atlassian Glossary

Workflow



Git - Command Line

Create a new repository on your computer
git init [project-name]

Copy existing repo from somewhere else to your computer
git clone [url]

Manage remote repository
git remote

Stage changes for commit
git add [file(s)] (use period . to add all files)

Take files from the staging index and save them in the repository
git commit -m “commit message”

Send changes to the remote
git push

Retrieve updates from the remote
git pull

Check the status of a repo
git status

Join two or more development histories together
git merge

Displays the difference between two versions of a file
git diff

Create an independent line of development
git branch (git checkout [branch] switches branch)

Review past commits
git log and git show

Installing Git

[Download link]

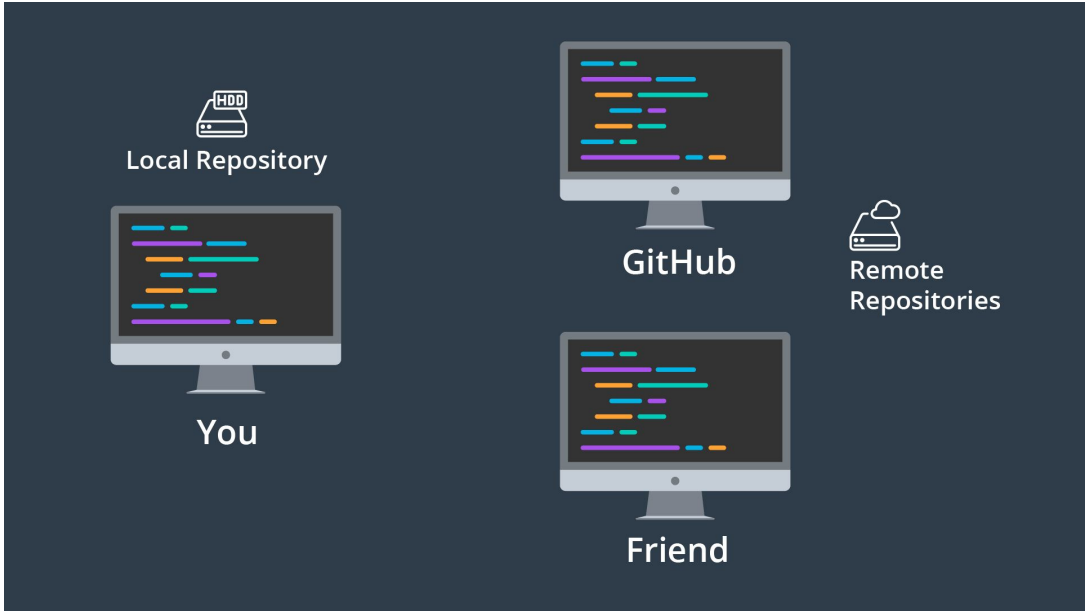
Confirm that you have Git installed by navigating to Program Files or using the “git” command in a CMD


```
git config --global user.name “John Doe”
git config --global user.email “johndoe@mail.com”
```

Remote Repositories

Since Git is a **distributed** version control system, there is not one main repository of information.

Each developer has a copy of the repository. You can have multiple remote repositories.



The **git remote** command lets us interact with the remote.

GitHub

Sounds similar to Git, but their purpose is quite different..

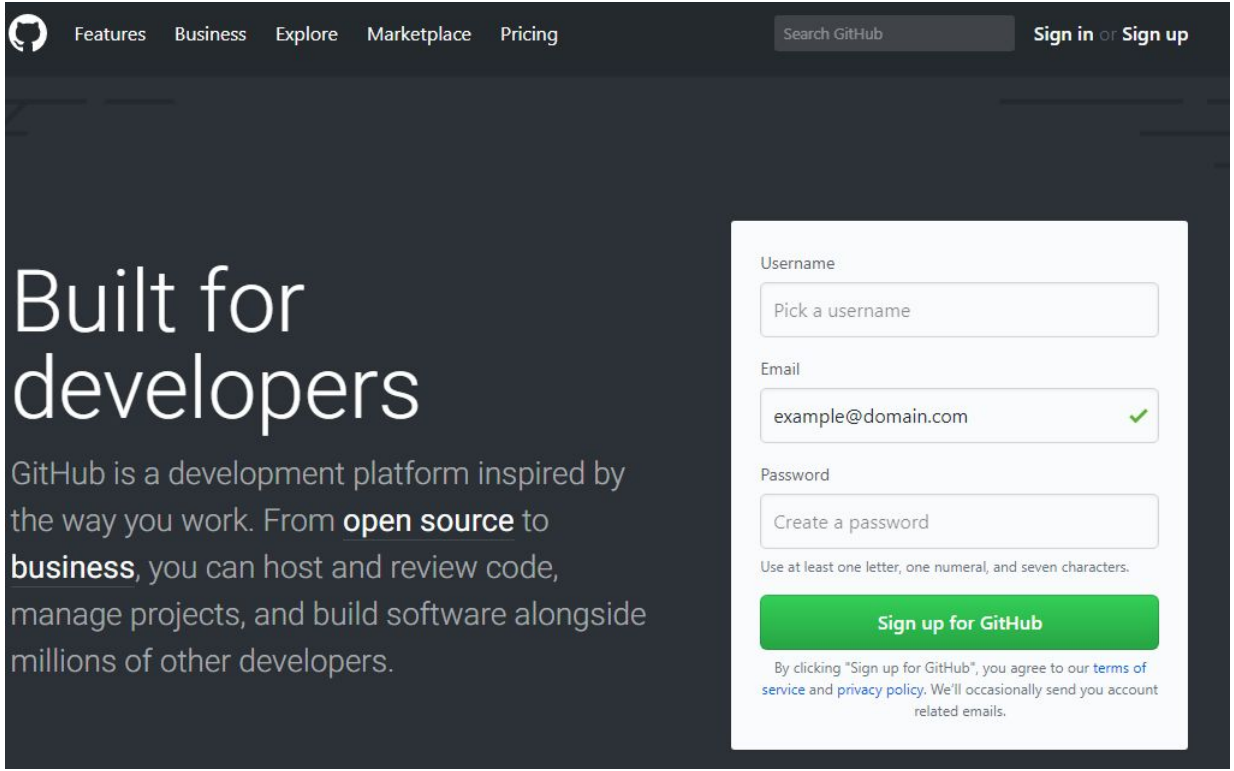
- Tool to manage Version Control Repositories
- Typically used in CMD
- Local Repository

- Service to host Version Control Repositories
- Typically used in the browser
- Hosts remote repositories

Creating a Github Account

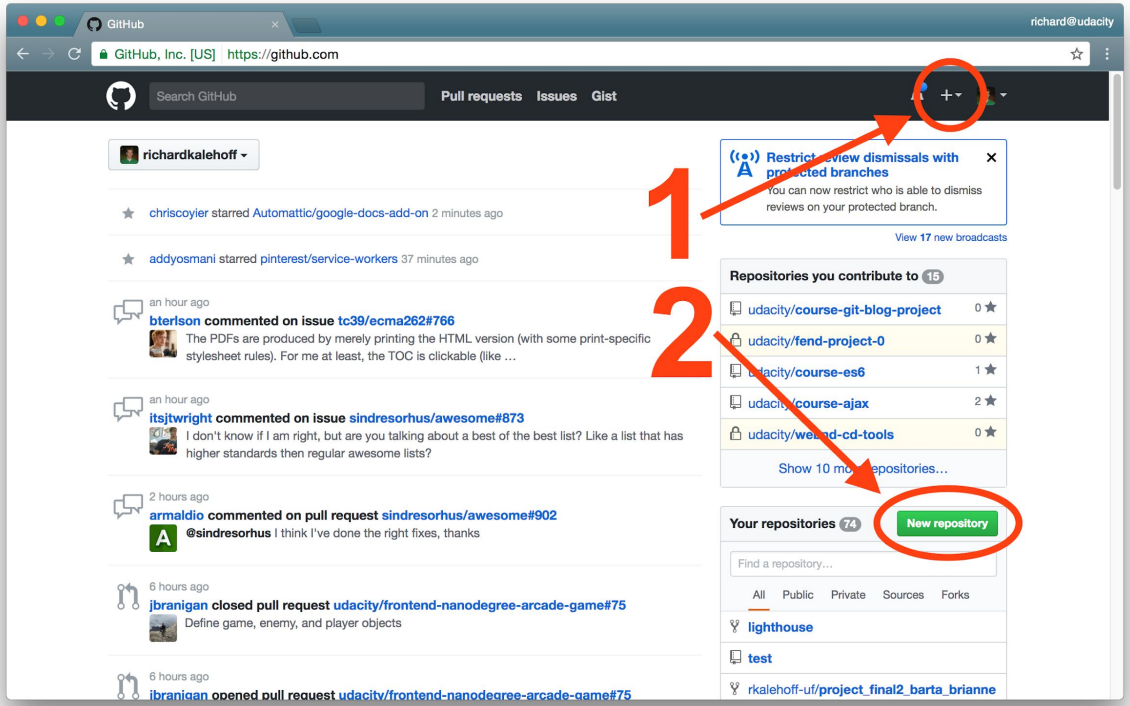
GitHub

https://github.com



Creating a new Remote Repository

GitHub

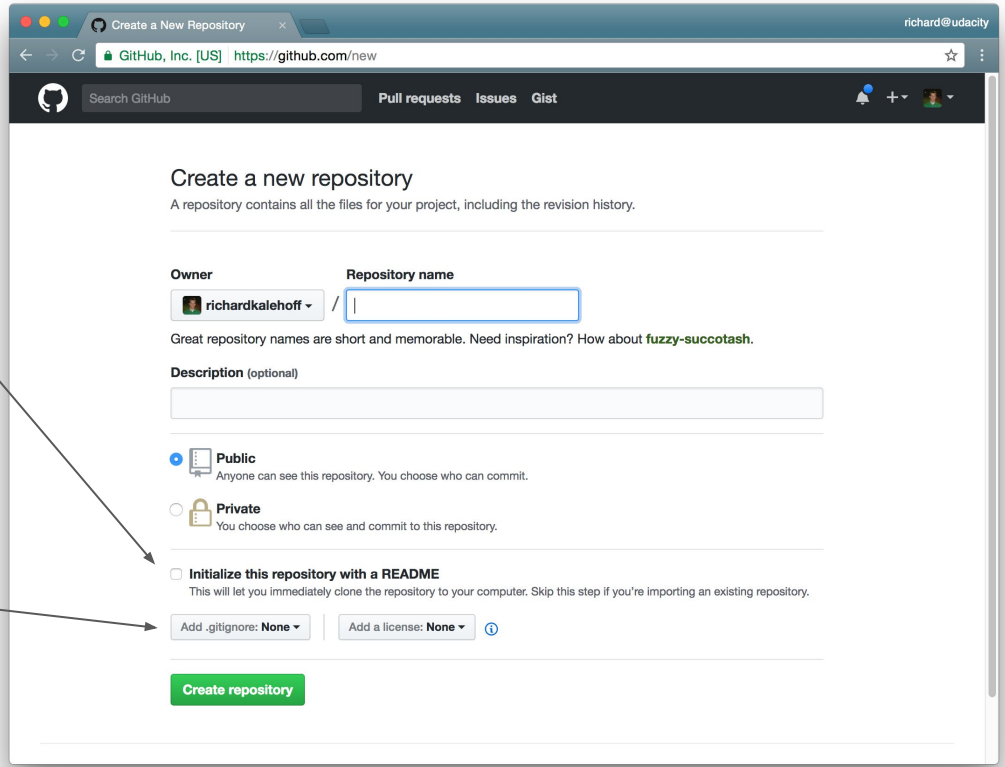


Creating a new Remote Repository

GitHub

What is a
README file for?

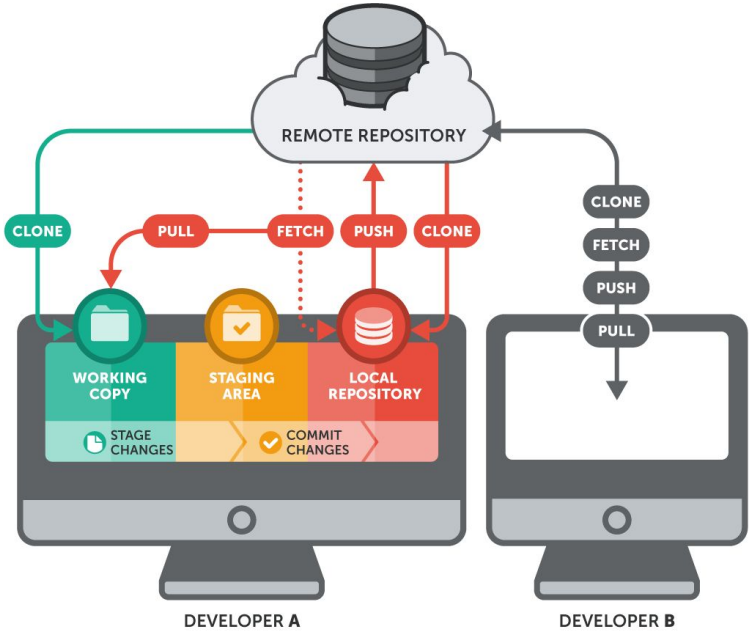
What is a
.gitignore file?



Connecting to the Remote Repository

git remote add origin https://github.com/[username]/[repository-name].git

git remote -v to verify that the remote repository is added correctly



Push Changes To a Remote

To send local changes to a remote repository you need to use the **git push** command:

git push <remote-shortname> <branch>

e.g. **git push -u origin master**

only used the first time you are pushing to a branch,
not required but helpful when **pulling** later

When do you want to push?

Pulling Changes From a Remote

To pull changes from a remote repository to a local one, you need to use the **git pull** command

git pull <remote-shortname> <branch>

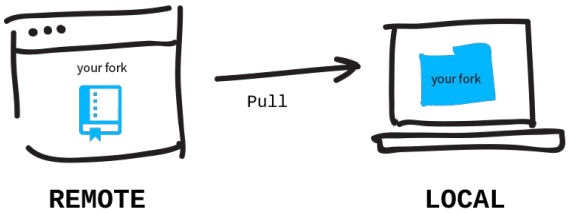
e.g. **git pull origin master**

When `git pull` is run, the following things happen:

git fetch
- the commit(s) on the remote branch are copied to the local repository
- the local tracking branch (`origin/master`) is moved to point to the most recent commit

git merge
- the local tracking branch (`origin/master`) is merged into the local branch (`master`)

When do you want to pull?



Checking your local repository:
git log --oneline --graph --decorate --all

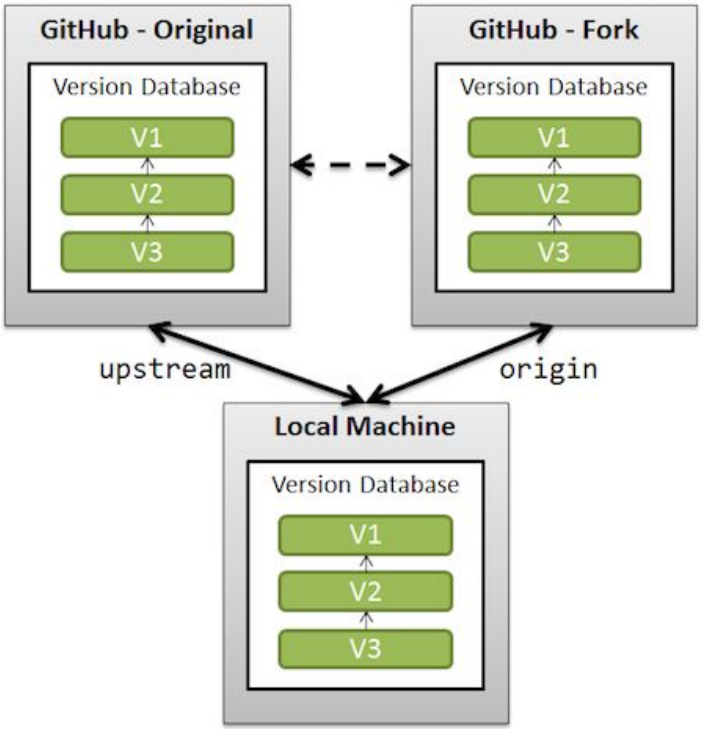
Forking a Repository

To fork a repository means to split it into an identical copy.

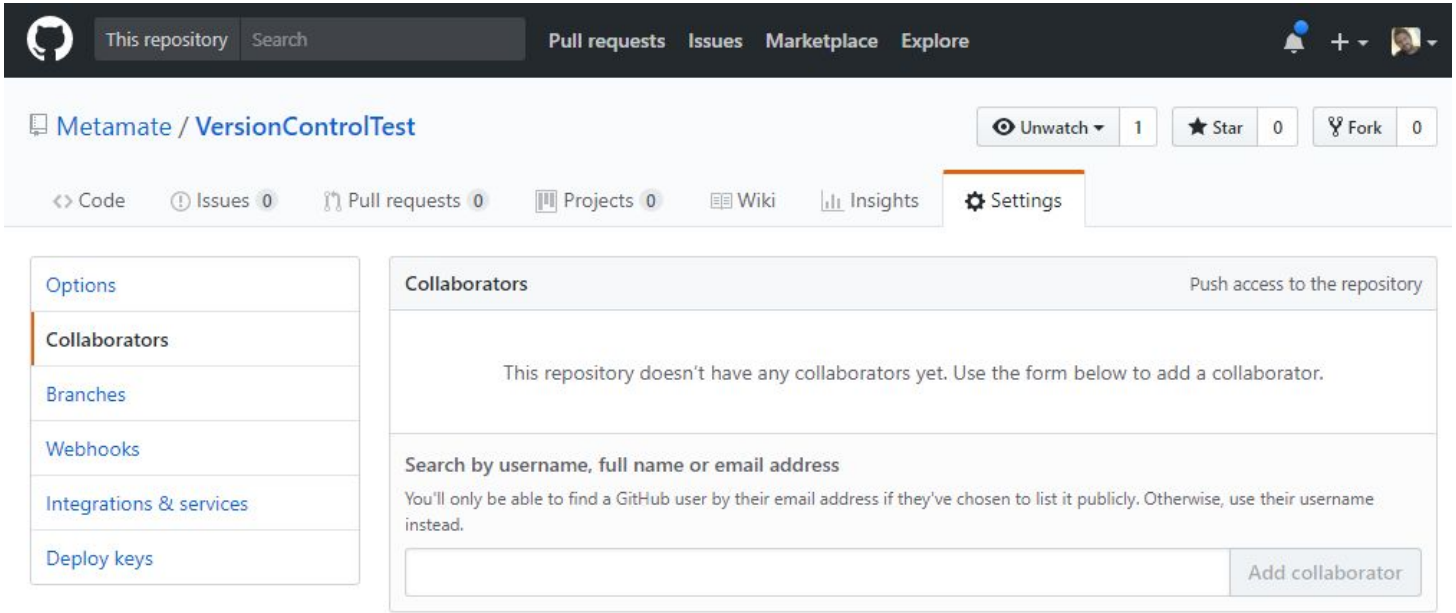
You have the control over this forked repository.
Modifying your forked repository does not alter the original repository in any way.

Forking is an action that's done on a hosting service, like GitHub.

If you have forked a project and you have code in your fork that's not in the original project, you can get code into the original project by sending the original project's maintainer a request to include your code changes. This request is known as a "**Pull Request**".



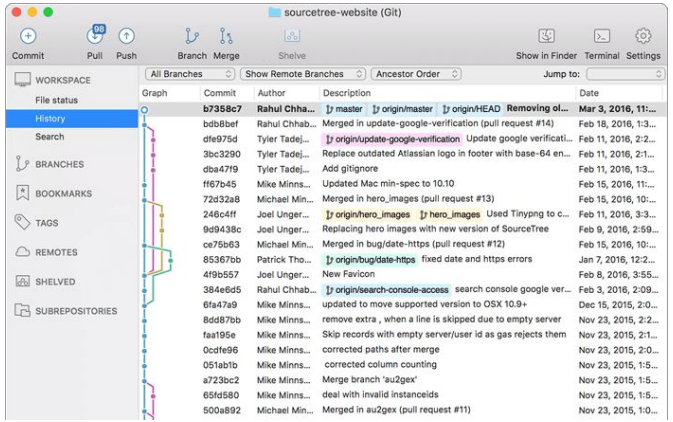
Collaboration setup



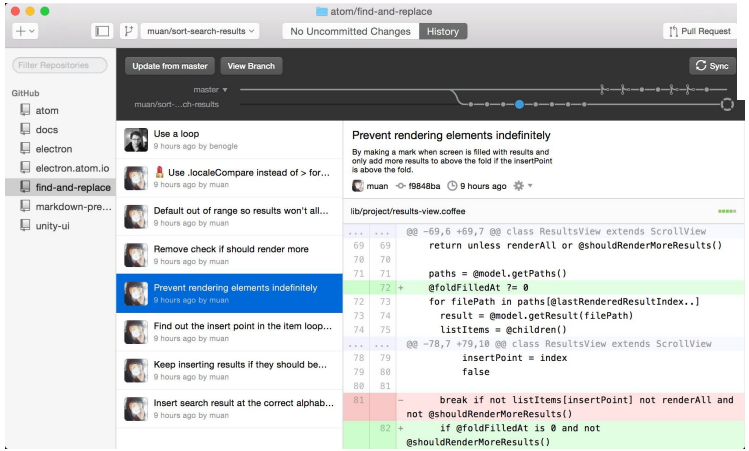
For collaboration, it's a very good idea to **use topic branches**.
Branches help isolate unrelated changes from each other

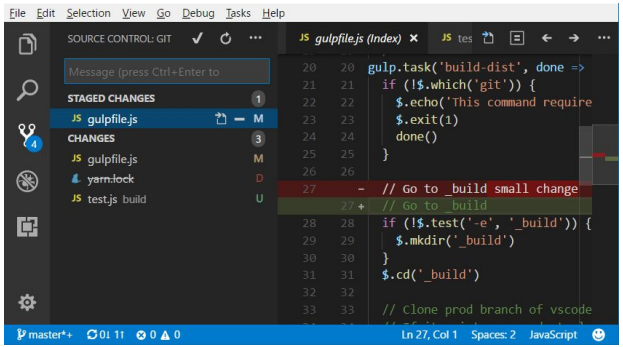
Sample Git UI Clients

Atlassian SourceTree



Git & VSCode





VSCode ships with Git support

Git & VSCode

[Pro Git Book]

[Resources to learn Git]

[Version Control in VSCode]