Git / Github

🎯 Goal:

- pull code from the encounter notes repo
- push code to the student code repo

What is wrong with Ada & Bob story?

- Lots of unnecessary files
- Difficult to track changes between versions
- Waste of time and resources — inefficient
- Error prone

What is Git?

https://git-scm.com

*"Git is a free and open source distributed **version control system** designed to handle everything from small to very large projects with speed and efficiency."*

- It is a fast implementation of **version control**
- It provides a **history** of content changes
- It facilitates **collaborative** changes to files

What does it allow you to do?

- It allows you to exercise **version control** on projects
  - keep **backups** of old versions
  - **track the history** of the changes of your code
  - **roll back** to old versions of the code

- It allows multiple people to work on different **features** of the product
- It allows you to **experiment** without fear of breaking anything

What is GitHub?

https://guides.github.com/activities/hello-world/

"GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere."

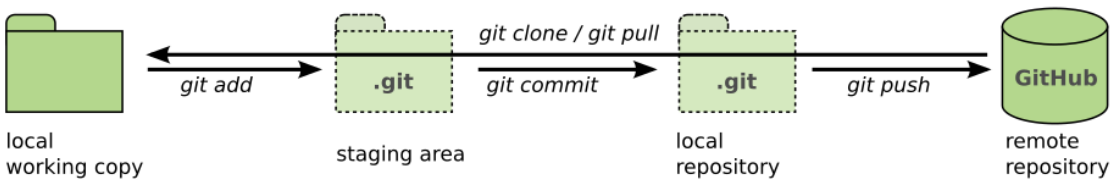
What does it allow you to do?

- It allows you to **collaborate** on projects
- It allows you to **publish** projects

A few concepts

- *Working Directory*: A local snapshot of the current status of your project. The directory as you see it at a given moment.
- *Staging Area*: Like a shopping cart, determining what you "commit" to the repository.
- *Local Repository*: .git directory inside your directory. The whole history of the project.
- *Remote Repository*: The whole history of the project on a remote server.

Basic git/github workflow



(A more verbose version of this step-by-step walkthrough is also available in the course materials.)

1. Create a *remote repository* on github

- Navigate to your github account, e.g. https://github.com/marijavlajic
- In the *Repositories* tab click on *New* and follow prompts:
  - repository name and description
  - README
  - .gitignore
  - license

3. Create a *local copy* of the repository
   - On your laptop navigate to your working folder, e.g. `spiced/week_01`
   - Clone the repository you created, `git clone <repo-url>`
     - Check `git status`

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

4. *Stage* changes *locally*
   - Create an empty file, `touch hello_cilantros.txt`
     - Check `git status`

```
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  hello_cilantros.txt

nothing added to commit but untracked files present (use "git add" to track)
```

   - Add the file to the staging area, `git add hello_cilantros.txt`
     - Check `git status`

```
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
```

```
  (use "git restore --staged <file>..." to unstage)
  new file:   hello_cilantros.txt
```

5. *Commit* changes *locally*
    - Commit to the local repo, `git commit -m "commit message"`
        - Check `git status`

```
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

5. *Push* changes *remotely*
    - Push to the remote repo, `git push origin main` (or `git push origin master` for older repos)
        - Check `git status`

There is an alternative approach of first creating a repo locally (with `git init`) and then uploading it to a remote repo, but the workflow described above is more similar to the one we will use for your encounter-notes and student-code repos

First-time / one-off git configuration commands

Sometime in August github started using *personal access token* instead of password authentication. Create your personal access token following the instructions here.

If you're using git for the first time on your computer, you might be shown some error message saying that you cannot make commits until you configure your git username and email address. Git associates your identity with every commit you make. You only need to do this one time; the commands you need to run to configure this are as follows:

1. `git config --global user.name "Your Name"`
2. `git config --global user.email "youremail@yourdomain.com"`

(to list global configurations use `git config --global -l`)

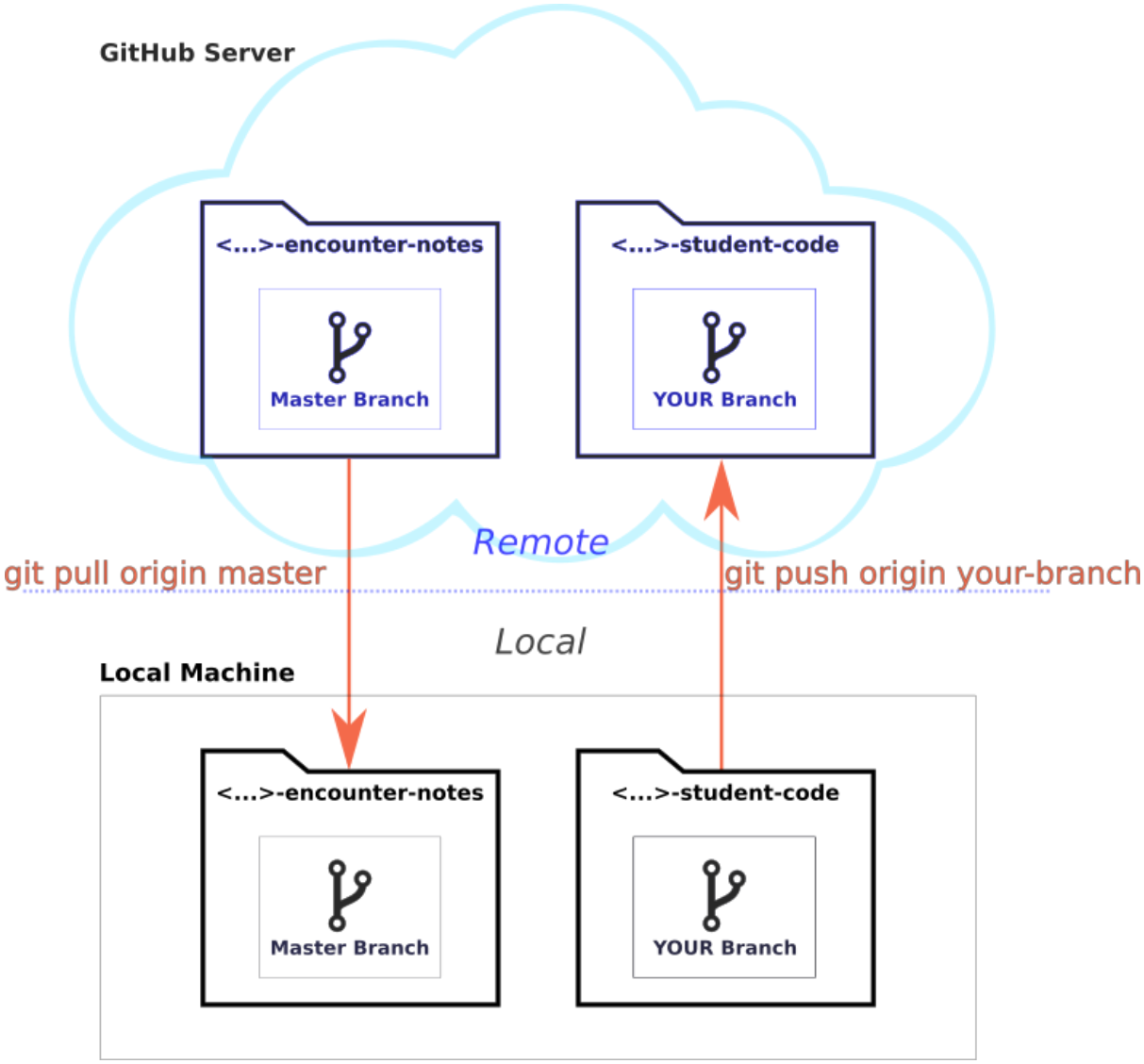
Important Note:

1. when prompted for your password on the command line, use your *personal access token*, as password authentication is deprecated
2. To stop the password prompt, you can use the following command when attempting to pull a private repo:
   `git clone https://<` *`your_personal_access_token`* `>:x-oauth-basic@github.com/<` *`user_name`* `>/<` *`repo_name`* `>.git`

Git/github workflow at SPICED

Two very important github repos:

- `scikit-cilantro-encounter-notes`
  - `git clone` https://github.com/spicedacademy/scikit-cilantro-encounter-notes if you haven't already done so
  - Teachers use this to make code/materials available to you
  - You will `git pull` from this repo
- `scikit-cilantro-student-code`
  - `git clone` https://github.com/spicedacademy/scikit-cilantro-student-code if you haven't already done so
  - You will use this to make code available to teachers
  - You will `git push` to this repo



1. Clone both repos (*one-off*):
   - `git clone https://github.com/spicedacademy/scikit-cilantro-student-code`
   - `git clone https://github.com/spicedacademy/scikit-cilantro-encounter-notes`
2. **Encounter notes** workflow (*daily/frequently*):
   - `cd scikit-cilantro-encounter-notes`
   - `git pull origin master`
3. **Student code** workflow:
   - `cd scikit-cilantro-student-code`
   - (*one-off*):

    - Create your own branch, `git branch <your-name>`
    - Checkout your branch, `git checkout <your-name>`
    - (You can combine the two commands above into `git checkout -b <your-name>`)
- (*daily/frequently*) `git add <your-file>`, `git commit -m "commit message"`, `git push origin <your-name>`

It is vital that you get into the habit of committing and pushing your work to your branch daily!