Johns Random Precourse stuff

John

Outline

What is the terminal?

When you turn your PC's brightness up so you can mine without having to place torches

Introduction to the Command Line

- Your computer is basically a machine that executes commands, you just have graphics to help visualize it
- You type everything, because thats how people did it back then.
- Sometimes called: shell, terminal, console (technically different, but you can ignore for now).
- To open it, open terminal in macos, or command prompt in windows.

Basics: Navigating Directories

- Basically file manager but no mouse.
- This is might not work on windows because blame microsoft.
- `cd` to change directories.
- `pwd` to display the current directory.
- `ls` to list files and directories.

```
$ cd Documents
$ pwd
/Users/yourname/Documents
$ ls
File1.txt  File2.txt  Dir1
```

Creating and Deleting

- note you can potentially remove all your files on the command line.
- great if you want to screw yourself over.
- `mkdir` to create a new directory.
- `touch` to create a new file.
- `rm` to remove a file.
- `rmdir` to remove an empty directory.

```
$ mkdir NewDir
$ touch NewFile.txt
$ rm NewFile.txt
$ rmdir NewDir
```

Viewing and Editing Files

- Basacily the equivalent of opening it in notepad.
- `cat` to display the contents of a file.
- `nano`, `vi`, or `vim` to edit files directly from the command line.

```
$ cat File1.txt
This is the content of File1.txt
$ nano File1.txt
```

But do I really need this?

- It is how you run python
- As a beginner, you can mostly get by by clicking the green button
- Sometimes green button no work, so its good to know how to do it from the command line
- Make sure you do not develop command line phobia.

Running Python Scripts

- Navigate to the directory containing your Python script using `cd`.
- Run a Python script with the `python` command followed by the script name.

```
$ cd path_to_your_script
$ python your_script.py
```

Using the Python Interpreter

- Very useful for playing around with concepts
- You can start the Python interpreter by simply typing `python` in the command line.
- You can execute Python commands directly in the interpreter.

```
$ python
>>> print("Hello, World!")
Hello, World!
>>> exit()
```

random tips

- Use `python --version` to check the installed Python version.
- Use `pip` to install Python packages: `pip install package_name`.

No I want to only click on green run button no terminal

- OK, well you have to use it when you ssh

What is ssh

- allows you to steal someone's computing power
- great if your own computer is a toaster
- but no gui, only shell
- its like someone gave you the keys to their car so you do not have to ride your tricycle.

How to ssh

- everyone gets access to their own virtual machine on the server
- `ssh username@somethingsomething`
- in theory you could use it to mine bitcoin

Demo if we have time

Outline

What is Git?

git
SUBVERSION
Final 2
Final Final
FINAL FOR REAL THIS TIME
OK This is definitely final
I prefer the real version control
I said the *real* version control
Perfection

What is Git (for real)?

- *Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers who are collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different computers).*
- Basically google docs for code.
- A command line program (but GUIs exist).
- It's how you download code off of github.

github

- git hub is the website, git is the actual program you use
- You can use a different source hosting site, but most just use github
- You can have a personal repo with your modifications by making a fork, which is done on github
- Github also has a bunch of random social media features too
- follow me on github btw

But why can’t I just use google drive?

Why use Git?

- Keeps a history of all changes.
- Branching and merging capabilities. (meaning two people can work on different parts of the code base at the same time)
- Distributed, meaning everyone that uses git has a copy of the source code, so you don't need internet to work on it.
- See who wrote what, and blame them for their bugs.
- If you don't use git you <u>will</u> get bullied.

Basic Git Commands

- `git init` Only need to do this once when making a new repo. If you download from github, you use `git clone`
- `git add` add to “staging”.
- `git commit` - Saves changes to the repository, with a small message saying what you did.
- `git push` - Uploads changes to a remote repository.
- `git status` See what you’ve added so far
- Lots of other commands, but we will only go over these.

Workflow

- Do some work
- Save in your editor
- Git add your changes, (sometimes you don't want to add all your changes)
- Git commit, and give a message
- Git pull to see if anyone else made any changes. If there are changes you need to merge them.
- git push to see it on github.
- Repeat

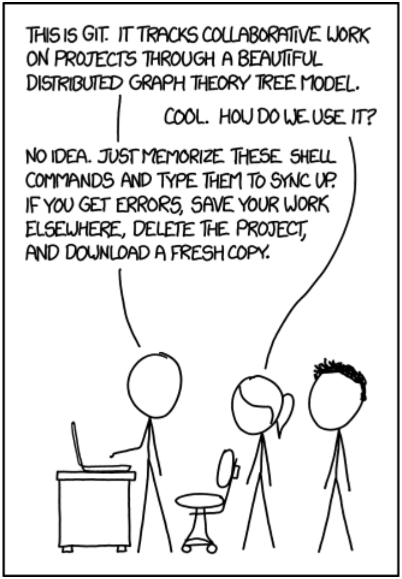
But that’s too hard

- Download a gui
- github desktop, gitkraken,
- vscode also has git integration
- Alternatively, edit directly on github (not recommended)

Faq

- Push didn't work! - Pull first
- Pull didn't work! - Commit first
- I committed but I still can't pull - You probably forgot to add before you commit
- I have a merge conflict! - Try not to panic, open the files, fix the errors, add and commit.
- I did commit, but now I'm in a weird editor - It's probably vim, type `i` to actually begin editing.
- How to exit vim? - esc `:wq`

Basically


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Interactive Example (if extra time)

Real life github

Outline

What is a bug?

- Basically, you are doing something wrong
- It's a lot easier writing wrong code than correct code.

Types of Errors

- Compile Errors
- Runtime Errors
- Logic Errors

Compile Errors

- Basically, you made a grammar mistake
- Examples: Missing semicolons, undeclared variables, mismatched brackets, typos
- Usually there will be a red squiggly line
- Relatively easy to fix, usually you copied something wrong.

Runtime Errors

- Occur while the program is running.
- Examples: Dividing by zero, trying to access an out-of-bounds array index, the famous null pointer exception
- Usually your program crashes.
- There is usually a line number to see where it failed

```
.10/site-packages/perlin_noise/perlin_noise.py", line
↪  50, in __call__
    return self.noise(coordinates)
  File
   ↪  "/nix/store/9zhyl4byxp5g895i07r8mdd0k15akcv4-python3-
   ↪  line 78, in noise
    return sum([
  File
   ↪  "/nix/store/9zhyl4byxp5g895i07r8mdd0k15akcv4-python3-
   ↪  line 80, in <listcomp>
    get_weighted_val(coordinates)
  File
   ↪  "/nix/store/9zhyl4byxp5g895i07r8mdd0k15akcv4-python3-
   ↪  line 62, in get_weighted_val
    return self.weight_to(coordinates) * dot(
  File
   ↪  "/nix/store/9zhyl4byxp5g895i07r8mdd0k15akcv4-python3-
   ↪  line 36, in weight_to
    def weight_to(self, coordinates: List[float]) ->
     ↪  float:
```

Logic Errors

- The code runs, but doesnt do what you want it to do
- Examples: Incorrect formulas, missing steps in a process.
- Often the hardest to detect because there are no explicit error messages.

Debugging Techniques

- Adding Print Statements
- Using Paper and Pencil
- Duck Debugging

Adding Print Statements

- Adding lines of code to display variables or messages.
- Allows you to see which code is being run, which code is not.
- Relatively simple, you learn hello world on the first day and that's all you need.

Using Paper and Pencil

- Write down variables and their values as you trace through the code.
- Allows for manual simulation of how the code runs.
- Sometimes writing stuff down just makes everything make sense

Duck Debugging

- Explain your code or problem out loud, as if to a rubber duck or inanimate object.
- Sounds really stupid, but sometimes works
- if you do this in public, you may get bullied

Other useful tips

- Google is your best friend
- Stackoverflow can also be helpful (they can also be mean)
- Maybe chatgpt (make sure you understand the code yourself)

Interactive example (if we have time)

- Examples: Making a sandwich
- Examples: Adding two numbers
- Examples: Doing laundry