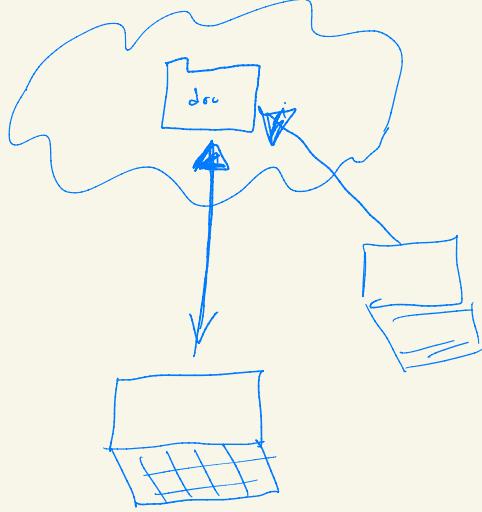


HAS Tools Notes

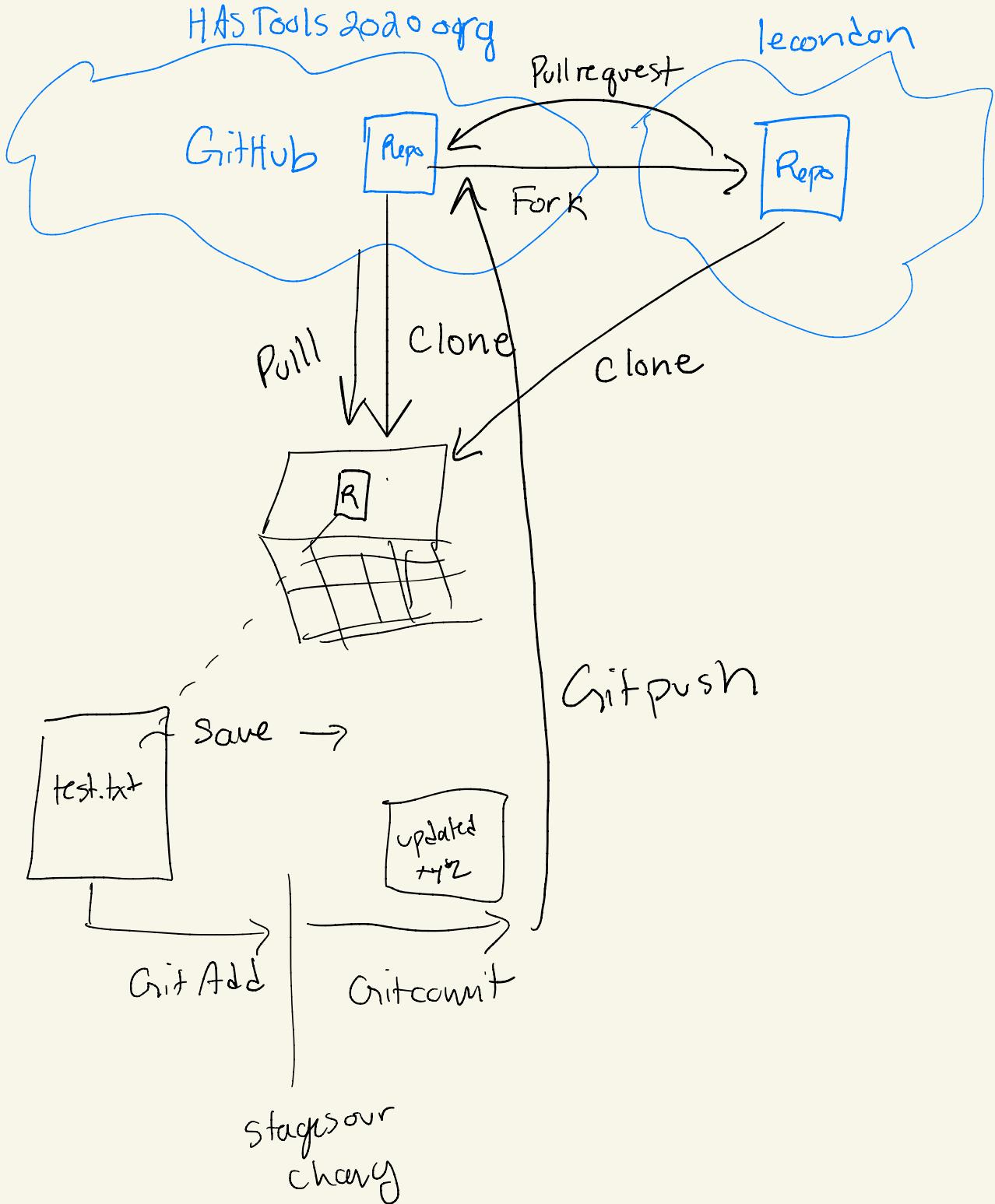
Fall 2020



Google docs / Dropbox / Box



HAs Tools 2020.org



Definitions from week 1

↙ command line

9-1-20

Terminal - Shell - Git Bash : place we give command directly to the computer

- language of computer (Bash shell (= google bash commands))

GUI - Graphical User Interface: Places where we point + click

Atom: Text editor (notepad, text pad (= other text editors)). It's the Best Github integration, designed for coding, lots of handy addons Super popular + widely used

Git: Version Control ⇒ tracking changes + making it possible for people to work together

Git Repo ⇒ directory of files with addition tracking features added (hidden .git files)

Github: Cloud server for git repos (github.com)

pull, push, clone ⇒ how we interact between local and remote repos
↑ ↑
on our computer on the cloud

Github Desktop: GUI for managing github repos, lets you do things like clone, pull, push

GitKraken: GUI for managing github repos - more powerful than Github desktop

Week 1 pitfalls

Common pitfalls:

- 1) Putting a repo inside a repo: Don't do it! if you do then make a fresh clone
- 2) Terminal - git commit and forget the "-m"
↳ control C to quit the operation in your terminal
↳ less prone to error if you commit through Atom or GitKraken
- 3) Don't do clone on one drive: do it locally instead
↳ No spaces in directory name

9-10-20

- Environments, packages & modules

↳ engine

Environment: A working "space" environment where you setup all of your software dependencies \Rightarrow lets you have multiple versions of things & software + packages + swap between them

Packages: A set of tools, A directory of python files with an "`__init__.py`" file
when we install a package we get this locally



Modules: A file of python code \Rightarrow define functions ~~and~~ classes & variables

Step 1: Activate environment

Step 2: Install package (only has to happen 1 time in your environment)

Step 3: Top of python script load package

Lists:

- Lists are 1D arrays [x, x, x, x]
- not data type dependent (int, floats, strings, boolean) ↳ but must be homogeneous with list
- Use []
- lists in lists [[]]
- indexing starts at 0
↳ indexing / slicing \Rightarrow pull out parts of our data structure
- * list comprehension
- Data structure
- Has its own methods = eg append

* function in python ()
method in python

function(x, arg1, arg2, ...)
object.method()

Control flow - Conditionals, for loops, while loops

- conditional statements: if else else if

if something is true
then do something

if its not true
then do something else

if the first thing is not true
but some second thing
is true then do ...

- Can as many nested conditionals as you want

↙ has to be something
with a T/F answer

- if <statement> :

L [Statement] I
indentation { S & }
matters! : { S 3 } ?

only happens
is true.

$$x = 5$$

if $x == 5$:

```
print("Here")
```

```
print ("done")
```

For loops:

- Repeats some action given a range of values
 - Can have as many nested as you want

range of values that will take

For index i in (range):
 Statement 1
 Statement 2
 Statement 3

in our for loop

```
for i in range(10):  
    print(i)  
  
print("done")
```

184)