Computer Session A/B

Python, R, and GitHub

 akenny430 akenny430 u/akenny430

Agenda

What we are going to cover today:

- Installing Python
- Installing R (and RStudio)
- Installing Git
- Setting up and using GitHub

We will be doing this for *both* Mac OS and Windows

- Python is easier to install on Mac OS
- R and Git are easier to install on Windows
- If you have a Linux machine, you probably don't need this tutorial

This whole tutorial will make use of your computer's terminal/command prompt

- Called the ***terminal*** on Mac OS/Linux
- Called the ***command prompt*** on Windows

We can use the terminal to open/run *any program* on our computer!

- Not what we are used to – there is a learning curve
- However, *very useful and important to know*
- ***VERY***

To access the terminal:

- Mac OS: using spotlight (Command + Space), type in "terminal"
- Windows: using the search bar, type in "command prompt"

Installing Python

Easiest way is to go to website: `https://www.python.org/downloads/`

- Latest version is `Python 3.9.7`

Mac OS: accept all defaults

Windows:

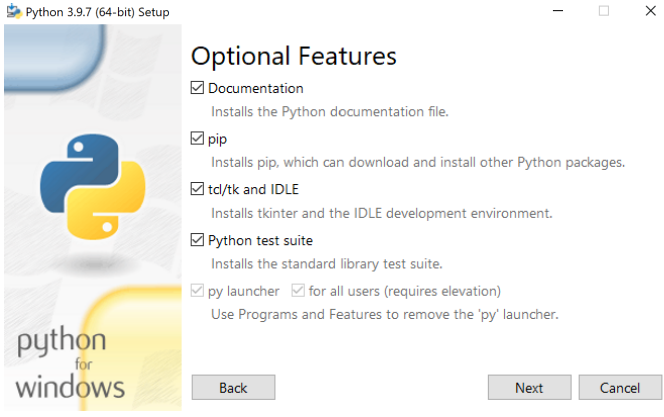
- Make sure Python is added to your *PATH*
- Choose "customize installation"
- Make sure that *pip* is being downloaded along with Python

Walk through of each now:

To check if each are installed:

- Mac OS: `python3 --version`
- Windows: `py -V`


Python 3.9.7 (64-bit) Setup
Install Python 3.9.7 (64-bit)
Select Install Now to install Python with default settings, or choose
Customize to enable or disable features.
Install Now
C:\Users\12016\AppData\Local\Programs\Python\Python39
Includes IDLE, pip and documentation
Creates shortcuts and file associations
Customize installation
Choose location and features
python
for
windows
Install launcher for all users (recommended)
Add Python 3.9 to PATH
Cancel


Python 3.9.7 (64-bit) Setup
Optional Features
Documentation
Installs the Python documentation file.
pip
Installs pip, which can download and install other Python packages.
tcl/tk and IDLE
Installs tkinter and the IDLE development environment.
Python test suite
Installs the standard library test suite.
py launcher
for all users (requires elevation)
Use Programs and Features to remove the 'py' launcher.
python
for
windows
Back
Next
Cancel

You can use Python by typing this into the terminal:

- Mac OS: `python3`
- Windows: `py`
- Type `quit()` to exit out of Python

Let's try it out!

Not the most efficient way to use Python

- We will look at other, better ways to use Python soon

Python by itself is a very general programming language

- We are more interested in statistics/data science/machine learning
- There are some things we want that *are not included* in base Python

However, there are other ***packages*** that can be installed to help us

- Packages add additional functionality that is not included in base Python
- Ways to read in datasets, make plots, do matrix multiplication, etc.

To install packages, we will use ***pip***

- `pip` should have been installed with Python
- Mac OS: `pip3 --version`
- Windows: `pip -V`

Relevant packages that we will want:

1. `numpy` (for general scientific computing, extremely powerful)
2. `matplotlib` (for making plots)
3. `sklearn` (for machine learning)
4. `pandas` (for data cleaning)
5. `jupyter` (gives us notebooks to use)

To install these, type: `pip install numpy matplotlib sklearn pandas jupyter`

- This might take a while, that is okay

There are other packages one can use as well

- Use pip to install those as well
- Type `pip list` to see a list of all packages already installed

(One of) the best ways to use Python is through ***Jupyter notebooks***

- An *integrated development environment* (IDE) that is good for Python
- Especially true for statistics/data science/machine learning

In terminal: `jupyter notebook`

- Yes, it has to be done through the terminal!

Walk through now:

Technically, Jupyter notebook files and Python files are different

- `.py` extension is single Python file
- `.ipynb` is Jupyter notebook file

Installing R

There are several steps, go to website: `https://cran.r-project.org/mirrors.html`

- Have to first choose a *CRAN mirror*, can choose any one from the United States (I chose Duke University)
- Latest version is `R 4.1.1`
- Click on appropriate link for Mac OS or Windows

For Mac OS:

- Depending on the type of Mac you have (Intel vs. Apple Silicon Chip), will be two different versions
- To check: click on Apple symbol in top left corner and choose "About This Mac"
- My Mac has an Intel chip

For Windows:

- There is only one option here

Walk through of both now:


Overview
Displays
Storage
Support
Service
macOS Big Sur
Version 11.3
MacBook Pro (16-inch, 2019)
Processor 2.3 GHz 8-Core Intel Core i9
Memory 16 GB 2667 MHz DDR4
Startup Disk Macintosh HD
Graphics AMD Radeon Pro 5500M 4 GB
Serial Number C02CR6FUMD6N
System Report...
Software Update...
™ and © 1983-2021 Apple Inc. All Rights Reserved. License and Warranty

We *could* use R from the terminal

- Mac OS: `R`
- Windows: more complicated, not worth our time

We will be using ***RStudio*** for R

- IDE that is designed around using R
- It is actually hard to use anything besides RStudio to run R (annoying)

To download, go to website: `https://www.rstudio.com/products/rstudio/download`

- Download the *free desktop version*
- Current version is `RStudio 4.1.1717`
- Same for both Mac OS and Windows

Walk through of both now:

RStudio comes with something called the ***console***

- It is used to run R commands without having to make an R script
- We use it to install packages and run quick commands
- By default, it is on the left side of the screen

With Python, we could use your computer's terminal to install packages and run commands

But with R, it is preferred to use RStudio's console instead

Base R actually has a lot of statistics functionality already built in

- R was made to do statistics, nothing else (at least originally)
- However, there are still packages made that make it even better

Relevant packages that we will want:

1. `ggplot2` (for making visuals)
2. `data.table` (for data cleaning)
3. `dplyr` (also for data cleaning, but similar syntax to ggplot2)

To install packages, we use use the RStudio console:

- Example, type `install.packages("data.table")`

To install Git, go to website: `https://git-scm.com/downloads`

For Mac OS:

- We first need to install *Homebrew*: `https://brew.sh`
- Then type `brew install git` into terminal

For Windows:

- Will automatically download as soon as you click the link:

Walk through both now:

To check installation: type `git version` into terminal

We can now use Git

- Mac OS: simply using the terminal
- Windows: using ***Git Bash*** (should have been installed when you installed Git)
- We can use the *same commands for both*

Note: finding the proper directory is harder on Windows

- You can look up the folder's properties for an exact location

We next have to set two global parameters for Git: the `user.name` and `user.email`

To set `user.name`:

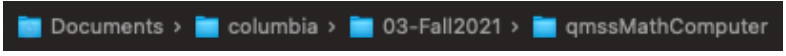
- Type `git config --global user.name "FirstName LastName"`
- Example: `git config --global user.name "Aiden Kenny"`

To set `user.email`:

- Type `git config --global user.email "youremail@whatever.com"`
- Example: `git config --global user.email "akenny430@gmail.com"`

Using the terminal/Git Bash, navigate to the folder where your project is located

- Use the `cd` command to move around different folders
- Example: `cd Documents/columbia/03-Fall2021/qmssMathComputer`
- The folder I want to be in is the `qmssMathComputer` folder



Then type `git init`

- This command *initializes* the local repository
- Check: type `ls -A` and there should be a `.git` file

Staging changes and making commits

Once initially created, every file in the folder is considered new

- To choose a file we want to "save", type `git add <FileName>`
- Example: `git add sampleNotebook.ipynb`
- This is called *staging a change*

To actually "save" the staged changes, we will *commit* them

- Type `git commit -m "Whatever message you want"`
- You can customize the message to be descriptive of the change you made to the file
- Every file that is *currently staged* will now be *committed*
- Example: `git commit -m "Initial commit"`

Setting up GitHub account and GitHub repository

Go to `https://github.com`

- Make yourself an account
- It does *not* need to be the same credentials we just used for `user.name` and `user.password`

The next step is to make a ***GitHub repository***

- This is where we will "post" all of the files in our repository
- Go to your GitHub profile and click on "Repositories"
- Then click "New"

You only need to give your repository a name

- Don't have to initialize a `README.md` or `.gitignore` file

Linking local Git repo to GitHub repo

We now have two repositories:

1. A local repo on our computer
2. A public repo on GitHub

We now want to *link* them together

To link the two repositories:

- The GitHub repo will have an *HTTPS* link you can copy
- Then type: `git remote add origin <CopyLink>`
- Example: `git remote add origin https://github.com/akenny430/qmssMathComputer.git`
- The first time you do this, you will have to sign in to your GitHub account

To check linking: `git config --get remote.local.url`

Now that our local repo is linked to the GitHub repo, we can "send" our files to GitHub

- This is called *pushing* our commits
- Think of it as "publishing" all of our saved work so far

Type `git push origin master`

- This will push every commit that we have saved so far

Try refreshing your GitHub repo!

The link for this GitHub repo is `https://github.com/akenny430/qmssMathComputer`

Summary

What we covered today:

1. How to access and use the terminal/command prompt
2. How to install Python and R
3. How to use them both in the terminal (not ideal) and with an IDE (better)
4. How to install packages for both to expand functionality
5. How to set up and use Git and GitHub

Keep practicing these skills over and over and over!!!

- It takes time and effort to become "fluent" in using the terminal, Python, R, and Git
- Being tech-savvy sets you apart when looking for jobs/internships/research
- There are a lot of great tutorials and documentation available online

Other comments

We decided to use Jupyter notebooks for Python and RStudio for R

- These are *not* the only IDE's you can use
- I personally use ***Visual Studio Code*** – look into it
- There are lots of options, shop around and find one you are comfortable with!

Using Git and GitHub is ***hard***

- There is a lot that goes into it besides the basics we covered
- Spend some time using them with some projects or class assignments
- My GitHub can be used for reference

In general, looking up errors on the internet will help a lot

- If you have a question, chances are it has already been asked (and answered)