APS106

UNIVERSITY OF TORONTO

The Coding Toolbox.

Week 1 | Lecture 2 (1.2)

if nothing else, write #cleancode

This Week's Content

- Lecture 1.1
  - Introduction
- **Lecture 1.2**
  - **The Coding Toolbox**
- Lecture 1.3
  - Variables, Operators, and Expressions

What is Programming?

- A way of telling a computer what to do.
- A computer can't infer (…yet).
  - Need to tell a computer every single step (or "instruction") it needs to do in a language it can understand.
  - How would you request an egg for breakfast to a chef and to a computer/robot?

- **To a Chef**
  1. Sunny-side up, please!

- **To a Computer**
  1. "Turn on stove"
  2. "Take out pan"
  3. "Take one egg out of fridge"
  4. "Crack egg"
  5. "Pour egg into pan"
  6. "Wait 5 minutes"

Programming is packaging instructions for our computer

(1) Python converts to machine code

- Python takes human-readable instructions and converts this to machine code (binary: 1s and 0s)

Most of this is outside the scope of APS106.

If we can provide the correct Python instructions in **Step (1)**, the computer will do what we tell it to!

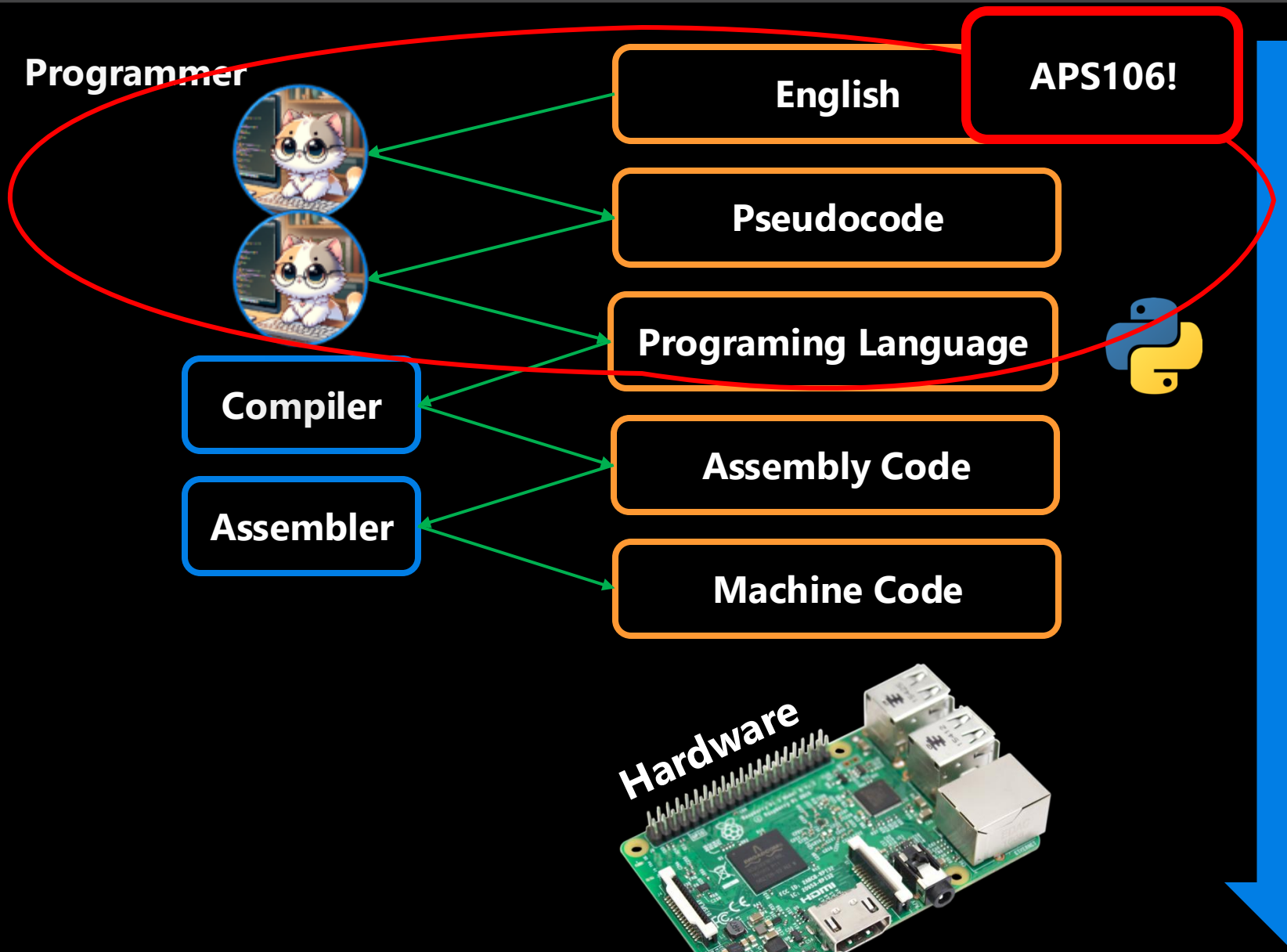
(2) Machine code determines transistor voltages

- This sequence of 1s and 0s determines which transistors in the CPU are ON and OFF, representing one specific instruction for the machine

(3) Instructions given sequentially on each clock cycle

- Each instruction is given in sequence and executed according to the processors clock speed (number of instructions per second)

How to Program a Computer.

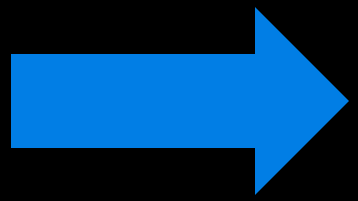


The power of programming languages

- `if x > 10:`

  `print("x is greater than 10")`

```
section .text
    global _start

_start:
    ; compare x with 10
    mov eax, [x]
    cmp eax, 10
    jle else_block ; jump to else_block if x <= 10

    ; if x > 10, print "x is greater than 10"
    mov edx, len_msg
    mov ecx, msg
    mov ebx, 1
    mov eax, 4
    int 0x80
    jmp end_if

else_block:
    ; code for else block goes here

end_if:
    ; code after if-else block goes here

section .data
    msg db "x is greater than 10", 0xa
    len_msg equ $ - msg
```



What do you need to code?

- A Computer
- An Integrated Development Environment (IDE)
  - A Programming Language (compiler or interpreter)
  - A File with an appropriate format (the program)

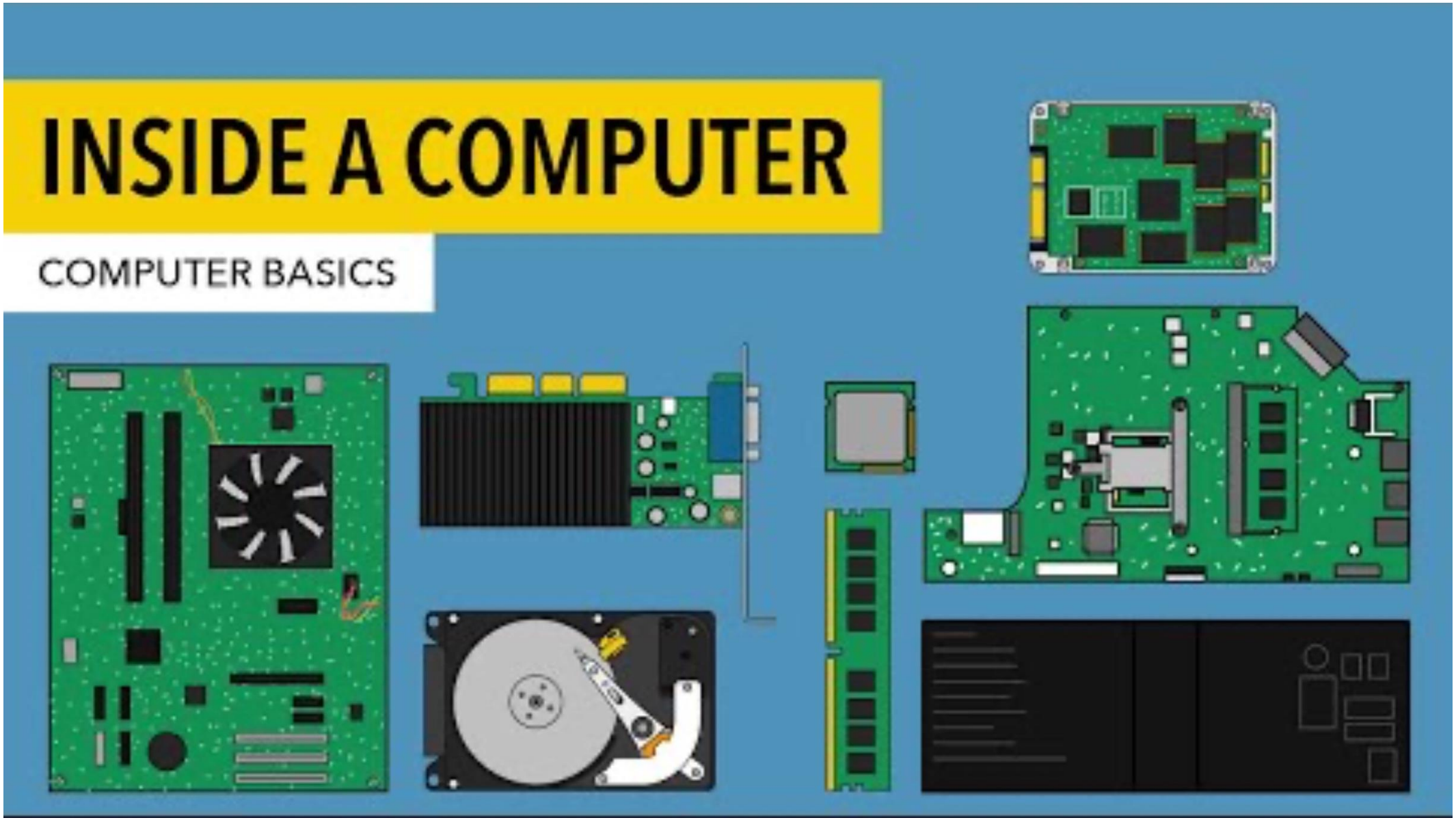
Outside Your Computer

Input Devices
- Keyboard
- Mouse
- Trackpad
- Touch screen
- Webcam
- Gesture trackers
- Eye trackers

Output Devices
- Monitors
- Printers
- Headphones
- Speakers
- Video/sound cards
- Braille readers

Eye Tracking Glasses

Braille Reader



Source: @GCFLearnFree YouTube Channel

Inside Your Computer

- Central Processing Unit
  - Runs the programs, including the operating system
  - Reads and writes the data between storage and memory
  - Composed of billions of miniature electronic switches called transistors
  - Popular examples include Intel's Core i7 and AMD's Ryzen 7

Inside Your Computer

- Storage
  - Disks, hard drives, thumb drives, etc.
  - Stores long-term data (files, programs, movies, etc.)
  - Non-volatile: maintains contents when powered off
- Memory
  - Random Access Memory (RAM)
  - Stores short-term data required for open programs and processes
  - Volatile: loses contents when powered off

Software programs are usually stored and executed from RAM.

Later this term, we will learn how to read from and write to long-term storage.

Integrated Development Environment (IDE)

- Text editor to write your program
- Compiler/interpreter to make the program executable

Integrated Development Environment (IDE)

- IDEs are programs that provide tools and features to programmers in a unified environment
- IDEs often include:
  - A code editor
    - A place to type and edit code, usually with colour-coded syntax highlighting to improve readability
  - Code compilers or interpreters
    - Turns the readable Python code into something the machine can understand
  - Debuggers
    - Pause the code at pre-determined locations and go line-by-line through your code
  - Version Control (i.e., git)
  - A built-in terminal for command line prompts
  - Integrated documentation (access to help resources from within the IDE)

Which IDE to use?

- IDLE is the official IDE included with Python that provides a basic environment for editing and running programs
- Other popular IDEs differ in features and supported programming languages
- Popular IDEs include:
  - Visual Studio Code (Microsoft)
  - Xcode (Apple)
  - Android Studio (Google)
  - PyCharm
  - Jupyter (web-based)
  - NetBeans
  - Eclipse

To write a document, you use a "Word Processor" such as Microsoft Word, Google Docs, or Apple Pages

Similarly, to write a software program, you use an IDE such as IDLE, PyCharm, or Jupyter

- This course will use both VSCode and Jupyter.
  - Jupyter will be used in lectures, and VSCode is the supported IDE in labs.

A Quick Note on File Formats and Extensions

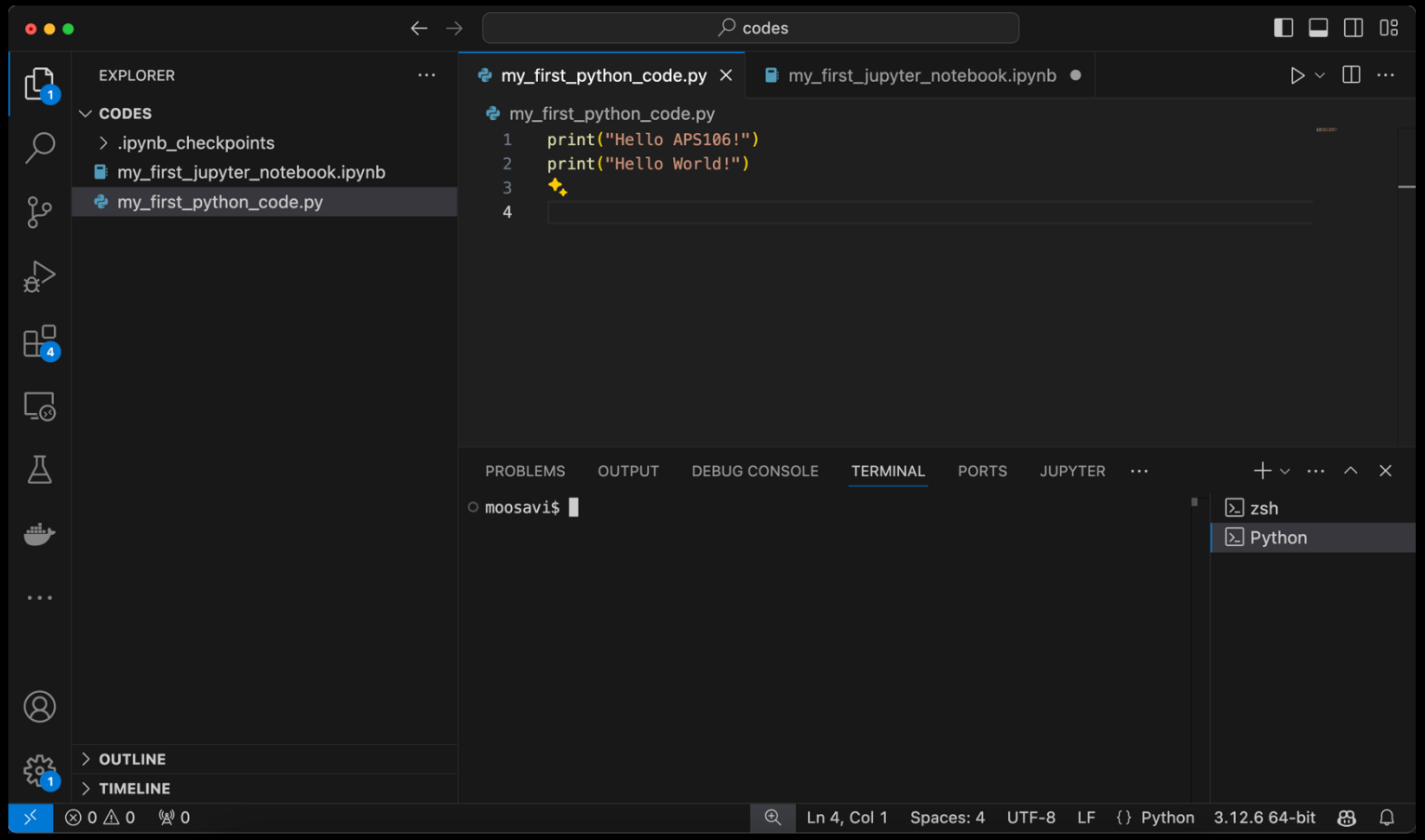
- Files store data in different formats based on specific definitions
- An "extension" is the file name's suffix, or the set of characters (usually 2-4) at the end of a file name, separated by a period (.)

File Types	Common Extension Examples
Plain text file	.txt
Word processor files	.doc, .docx, .pdf
Image files	.jpg, .gif, .png
Compressed/Archive files	.zip, .rar, .7z
Video files	.mpg, .mp4, .avi
Executable files	.exe, .bat, .bin
Python file	.py
Jupyter Notebook file	.ipynb
Comma Separated Value file	.csv

File extensions help indicate:
- The file format
- How the data is structured
- Which programs can open it

Running Your First Python (.py) File

- Make a folder in your machine to store today's lecture codes
- Open the folder in VSCode
- Type your first code!
- Run the code



Introducing Python

- No end-of-instruction separators, such as semicolons (like in C or Java)
- **Python programs are stored in .py files**
- Comments start with a # character
- **Whitespace matters (exactly 4 spaces means indentation)**
- Python is an interpreted language (not a compiled one)
  - You can run code one statement at a time, just like a calculator or Matlab
  - This means variables can change type during runtime, and do not have to be declared before running

Anaconda Navigator

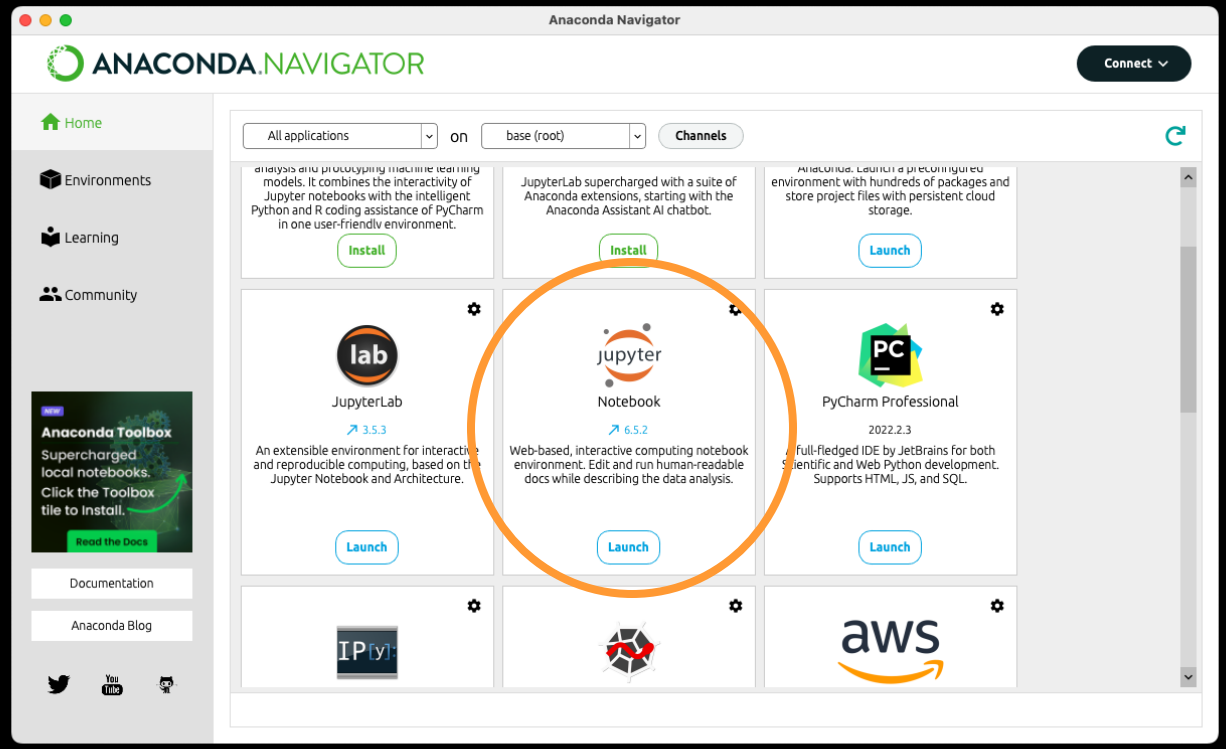
- Anaconda is a distribution of Python that includes tools and packages geared towards scientific computing (such as data science and machine learning)
- Anaconda Navigator is the graphical user interface (GUI) allowing users to install and manage their programming environment **without command line (terminal) prompts**

Anaconda Navigator

- Your first core use of Navigator is to open Jupyter Notebook (not JupyterLab)

More advanced (optional):

- In the Environments tab on the left you can manage the packages installed for different "environments"
- Installing a specific version or package for one project shouldn't necessarily affect all projects
- "Environments" allow developers to isolate project workspaces (with specific Python versions and installed packages



What is Jupyter?

- Jupyter Notebook 

  - Interactive and web-based environment
  - Creates "notebooks" (.ipynb files) that can combine live code, visualisations, and narrative text
  - Code can be divided into individually-executable "cells"
  - Cells can include either executable code, or formatted text and images
  - Can export notebooks to HTML or PDF

Opening Your First Notebook (.ipynb) File

- Open Jupyter Notebook through Anaconda Navigator
- Jupyter Notebook will open in your web browser, and you will see the directory (folder) system for your computer
- Navigate to a folder where you would like to work on your first juypyter notebook

Let's Quickly Explore Jupyter

- Explore different cell types
- Execute Python code in Jupyter
- Other helpful Jupyter tips

APS106

UNIVERSITY OF TORONTO

The Coding Toolbox.

Week 1 | Lecture 2 (1.2)

if nothing else, write #cleancode