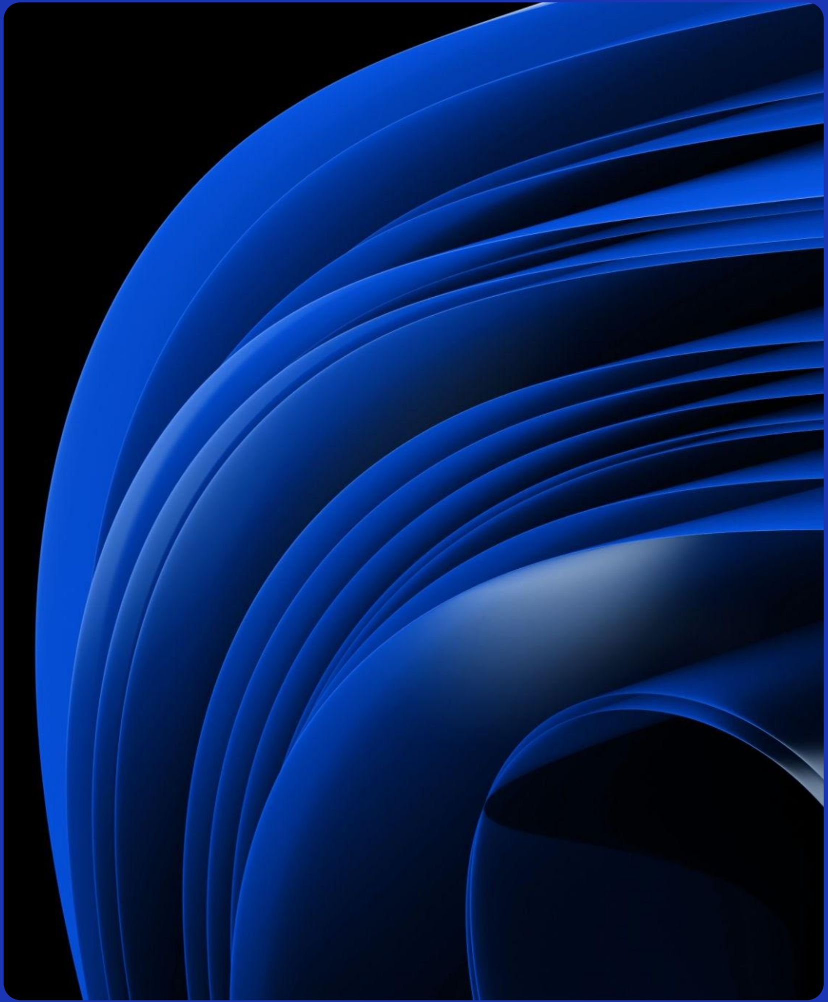
A My Path MN and BDPATCF Collaboration

Intro to Python

Led by William Munnich
Friday 3/22/2025



Review

Variables

- 🔹 *Variables* are like labeled jars where we store information.
- 🔹 They help us keep track of values like names, numbers, or anything else in our programs.

Integers & Floats

- 🔹 *Integers* are whole numbers (like 3, -7) and *floats* are decimal numbers (like 4.5, -0.1).
- 🔹 We use them to do math in Python.

Review

Booleans

🔹 *Booleans* are either `True` or `False`.

🔹 They help our programs make decisions by answering yes/no questions.

Strings

🔹 *Strings* are text – like words, sentences, or even emojis!

🔹 They're written with quotes like `"Hello"` or `'Python 🐍'`.

Review

Lists

- 🔹 *Lists* are like boxes that can hold many items – numbers, words, or even other lists.
- 🔹 They let us group things together in one place.

Dictionaries

- 🔹 *Dictionaries* store data in pairs, like a word and its meaning.
- 🔹 You look things up by using keys (like names or labels).

Review

Basic if

- *If statements* let us make choices in our programs.
- We can say, “If something is true, then do this.”

if → elif

- *Elif* means “else if.”
- It lets us check more than one condition in a row to decide what to do.

Review

if → elif → else

- *Else* is what happens when none of the previous conditions were true.
- This makes our programs more complete by covering every possibility.

for

- *For loops* let us repeat things a set number of times.
- They're great for going through lists and doing something with each item.

Review

while

- *While loops* keep going **as long as** something is true.
- We use them when we don’t know how many times we need to repeat something.

Review: Python Error: *SyntaxError*

Error Name	Description	Example	Common Causes	Suggested Fixes
SyntaxError	Occurs when code violates Python's syntax rules, like missing colons or parentheses.	`if x = 5: print(x)` (missing == for comparison)	Typos, missing punctuation, incorrect indentation.	Check error messages for line and caret location.

python:

```python
if True
      print("Hello")
```

Terminal Output:

```
 File "<stdin>", line 1
    if True
          ^
  SyntaxError: expected ':'
```

python:

```python
  print("Hello"
```

Terminal Output:

```
   File "<stdin>", line 1

      print("Hello"

            ^

  SyntaxError: '(' was never closed
```

python:

```python
if x = 5:
      print("x is 5")
```

Terminal Output:

```
 File "<stdin>", line 1
    if x = 5:
         ^
  SyntaxError: invalid syntax
```

Review:Python Error: *IndentationError*

Error Name	Description	Example	Common Causes	Suggested Fixes
IndentationError	Raised when indentation is inconsistent, critical for Python's block structure.	`if True: print("Hello") else print("World")` (missing indent after `if`).	Mixing tabs and spaces, incorrect nesting.	Use formatters like Black, ensure consistent 4-space indentation.

```python
def say_hello():
print("Hello")
```

Terminal Output:

```
File "script.py", line 2
    print("Hello")
    ^
IndentationError: expected an indented block after function definition on line 1
```

```python
print("Start")
        print("Why am I here?")
```

Terminal Output:

```
File "script.py", line 2
    print("Why am I here?")
    ^
IndentationError: unexpected indent
```

Review: Python Error: *TypeError*

Error Name	Description	Example	Common Causes	Suggested Fixes
TypeError	Raised when an operation is applied to an inappropriate type.	`"hello" + 5` (trying to add string and integer).	Incorrect type usage, mismatched function arguments.	Ensure correct types, use type conversion, check documentation.

```
num = 5
text = "hello"
result = num + text
```

```
number = 123
print(number[0])  # Integers don’t support indexing
```

Terminal Output:

```
File "script.py", line 3, in <module> result =
num + text TypeError: unsupported operand
type(s) for +: 'int' and 'str'
```

Terminal Output:

```
File "script.py", line 2, in <module>
print(number[0]) TypeError: 'int' object is not
subscriptable
```

Review: What If There is No Error But it Just Doesn’t Work? Debugging!

When in doubt, use print statements in between. 👍

```
total = 0
for i in range(5):

    total += i
print(f"Final is: {i}")

#prints
    Final is: 5
```

```
total = 5
for i in range(5):
    print(f"Loop iteration: {i}")
    total += i
print(f"Final is: {i}")

#prints
    Loop iteration: 0
    Loop iteration: 1
    Loop iteration: 2
    Loop iteration: 3
    Loop iteration: 4
    Final is: 5
```

Review: Pseudocode

-Not required but it's good practice and has benifits
-Written in pure comments if in a program file or HOWEVER is clear to you
-For clarity
-Planning things out
-Maybe you don’t know how to program it yet but you know how conceptually it will work

Review: Functions

A block of code that can be reused over and over again.

```
def greet():

        print("Hello!")


greet()
```

```
greet(name):

        print("Hello", name)


greet(“James”)
```

```
def greet_user(name, hour):
    if hour < 12:
        time_of_day = "morning"
    elif hour < 18:
        time_of_day = "afternoon"
    else:
        time_of_day = "evening"

    print(f"Good {time_of_day}, {name}!")


greet_user("Ava", 9)      # → Good morning, Ava!
greet_user("Liam", 15)    # → Good afternoon, Liam!
```

Review:Referencing Functions

You can **reuse functions from another Python file** by importing them, like this:

```
from my_file import my_function
```

It’s just like referencing a function **someone else wrote** — which is exactly what happens when you import a **library** like math, csv, or random.

You’re using **pre-written code** to save time and avoid writing everything yourself!

Review: File Types & File Extensions

Scripting	.py, .ps1, .sh
Simple Data Storage	.csv and .json
Audio	.mp3 and .mp4
Word Doc	.doc and .docx
Database	.db
Querying a Database	.sql
Markup	.html, .md

Review: What Is a Database?

Key Features of a Database:

- **Stores lots of data** (names, links, numbers, etc.)
- **Keeps it organized** (tables, rows, columns)
- **Makes it easy to search and filter**
- **Can be used by apps, websites, and games**

Real-world Examples:

- Instagram uses databases to store user posts, comments, and likes.
- A video game stores player stats and scores in a database.
- Schools use databases to keep track of students and grades.

What is Web Scraping?

🌐 Definition:

- Web Scraping is **automatically gathering information from websites** using code.

🎯 Simple Analogy:

- “Imagine copying and pasting information, but **faster, automatic, and smarter!**”

Ethical and Legal Rules ⚖️

- ✅ **Always respect** website rules (`robots.txt` file).
- ✅ **Only scrape publicly available information** (e.g., Wikipedia, NASA, public APIs).
- ❌ **Never scrape** sensitive personal data, copyrighted materials, or private content.

Tools for Today's Lesson 🛠️

requests

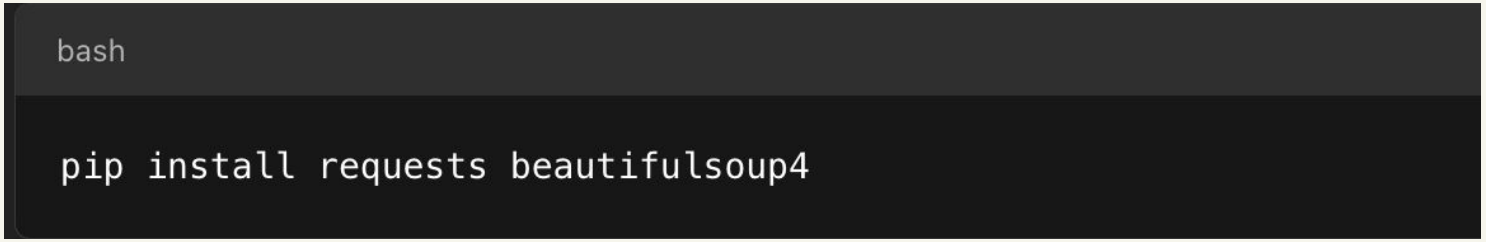
- **Downloads webpages** directly into your Python program.

BeautifulSoup

- **Extracts and organizes** exactly the parts of the webpage you want.

Installing the Tools (Quick Demo)

Run this command in your Terminal or Command Prompt:



```bash
pip install requests beautifulsoup4
```

In-Class Project 1 (Fully Completed):

Title: "Scrape and Save Simple Wikipedia Information" (Plug-and-Play)

Steps:

- Fetch a Wikipedia page about "Python (programming language)."
- Use BeautifulSoup to grab the text from the page.
- Save text into a `.txt` file.
- Convert it into `.csv`.
- Delete the original `.txt` file.

In-Class Project 2 (Partially Completed - Simple Fix):

Title: "Scraping Current Weather Info" (Weather website)

Hint/comment: “Uncomment the lines and run. If errors appear, use ChatGPT or Stack Overflow to troubleshoot.”

In-Class Project 3 (Intermediate Difficulty):

Title: "Extracting Audio or Video Links"

Scrape links to freely available video/audio files (e.g., a podcast homepage)

You will:

- Uncomment selection code
- Write loop to print or save these links to a file.

Class Discussion & Troubleshooting Practice (10 mins):

- Encourage students to use AI to debug problems.
- Demonstrate asking a simple question to ChatGPT, example:
    - "My BeautifulSoup select isn't working, what did I do wrong?"

Take-Home Project (Low Barrier, Higher Autonomy):

Title: "Create Your Own Web Scraper!"

Choose one:

- Scrape headlines from a favorite free news/blogging site.
- Scrape free stock price info from a financial site.
- Scrape open, free podcast episodes.