Importing Python Code and an overview of Project 3

April 26, 2021

Administrative Notes

Project 2 is due tonight!

Lab 12 is tomorrow - more on the next slide

Project 3 is not yet ready; it will be released as soon as I finish it

Reminder: the final is May 17, 3:30 - 5:30 pm on Blackboard - let me know if there’s an issue with that time & date

A note on lab 12

Lab 12 tomorrow addresses some of the same issues that we’re going to be talking about today…

Virtual environments - they let you set up a software based environment that makes it look like you’re running on a hardware-based machine by yourself. Several advantages:

- You can run different programs and guarantee they won’t interfere with each other
- You can buy expensive hardware and make sure that it’s used to the maximum efficiency
  - I don’t mean to be negative, but your laptop and mine are mostly “wasted” because they’re not being used all the time.

In Python

A “virtual environment” lets you configure different projects to run using different versions of the Python interpreter; different libraries and packages; different toolsets; etc.

- You don’t have to update all the Python work you’ve done just because a new and incompatible change is made by somebody somewhere to some code you rely on.

Importing and using code in Python

Available Code

One of the big advantages of using Python is that there is a ton of code that exists “in the wild” that is available for your use.

- Math library - https://docs.python.org/3/library/math.html - math functions
- Numpy - https://numpy.org/ - arrays; linear algebra; FFT; random numbers; …
- Pandas - https://pandas.pydata.org/ - data analysis/data science
- Pillow - https://pypi.org/project/Pillow/ - image manipulation
- Matplotlib - https://matplotlib.org/index.html - graphics and plots
- Ggplot - http://ggplot.yhathq.com/ - more plots

See https://www.ubuntupit.com/best-python-libraries-and-packages-for-beginners/ for more

Importing Code

You can import any of these existing libraries into your Python program

You can also import any code you’ve previously written yourself (or that your professor or classmate has written)

If the package has already been installed, just use the import statement:

```
 import math  #imports the existing math library; you can now use any of the
              #functions in that library
 import hailstone #imports the hailstone.py program from last week and lets you
                  #use the “flight” function
```

There are multiple formats

Basic:

```
import math
```

Means you must include the module name in each function call:

```
x = math.sin(0)  # if you just said “sin” the Python interpreter wouldn’t know what you meant
```

Rename:

```
import math as m
```

Now you can call the functions with a simpler name: x = m.sin(o)

Import formats

Only import the functions you need:

```
  from math import sin
```

Then you don’t have to refer to the module - x = sin(0) # valid

Or import all functions with the * wildcard character:

```
   from math import *
```

Now you can refer to all the math functions without referring to the module

Just make sure you don’t have another function with the same name!!!!

An example using the pandas module

You can read in and process a .csv file with a single statement:

First, install pandas

Then in your program:

```
import pandas as pd
```

Now read in the file:

```
df =  pd.read_csv("Spring_2022.csv")
```

And we can easily drop those pain-in-the-neck rows of commas:

```
df.dropna()
```

Importing your own code

We’ll write a program called “dates.py” that includes a function called “convert_date.”

Then we’ll write a separate program that imports “dates” and calls that function.

The separate program is:

```
import dates
if __name__ == “__main__”:
        print(dates.convert_date(‘07042020’))
```

```
from dates import *
if __name__ == “__main__”:
    print(convert_date(‘07042020’))
```

That’s it - really!!!

‘The rules’ for importing your own code:

1. The name of the module MUST end in ‘.py’
2. The module must be in the Python path so that it can be found
    a. Be in the same directory as the calling program
    b. Be in a directory that Python always searches for programs
    c. Include the entire path to the file in the import statement

The Jupyter Notebook

https://jupyter.org/

Formerly the “iPython Notebook”

- A format for producing a document including Markdown language, executable code, and code results
- An ordered series of cells. You specify the format of each cell
- “A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation slides, LaTeX, PDF, ReStructuredText, Markdown, Python) “