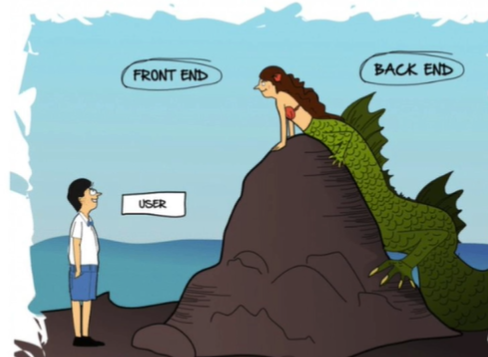
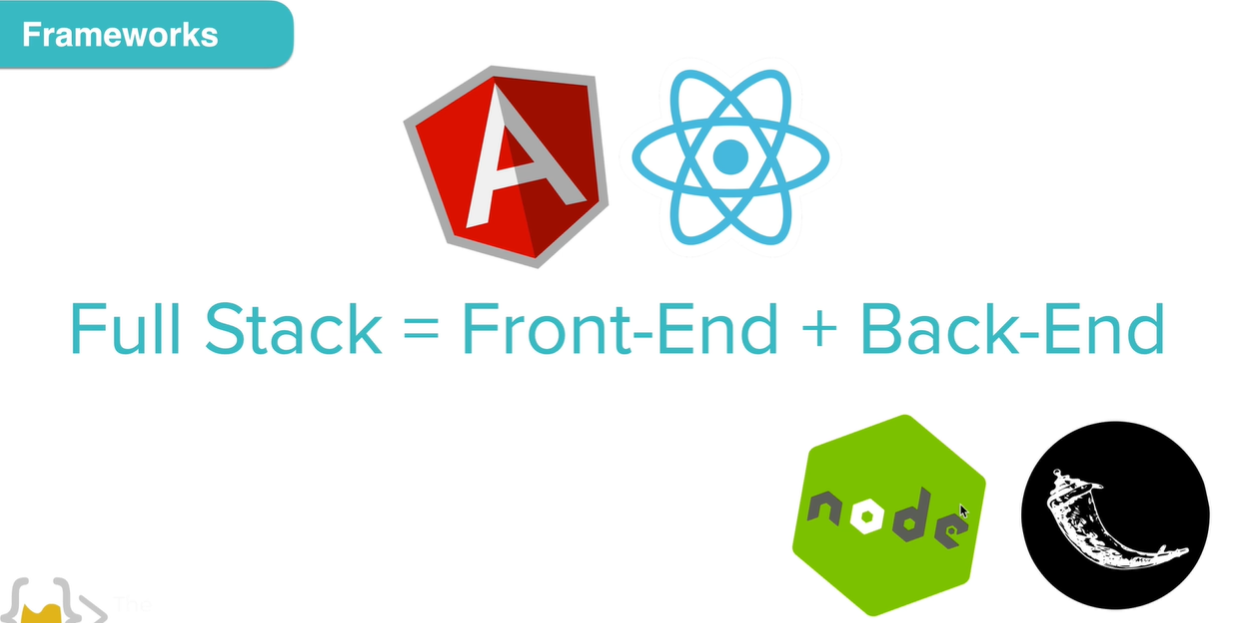
Day 54 web dev in python







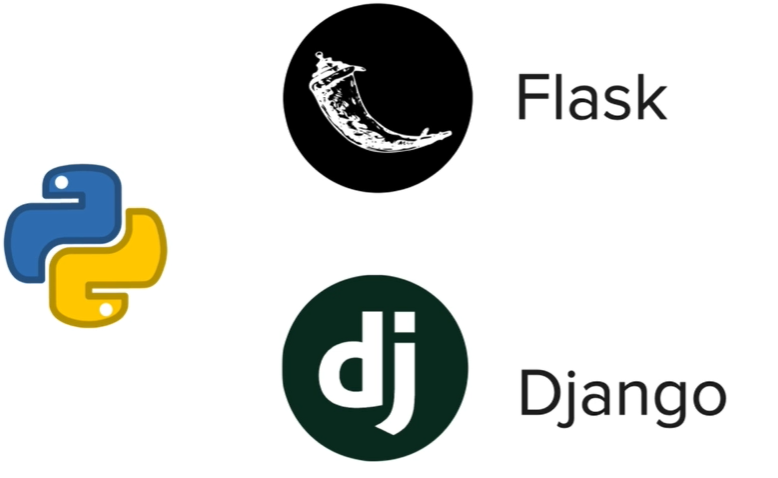
Frameworks tools that come with a lot of code prebuilt for common functionalities that you find when building the front and back end.

Python backend frameworks include Flask, Django, Bottle, Cherry pie, pyramid,

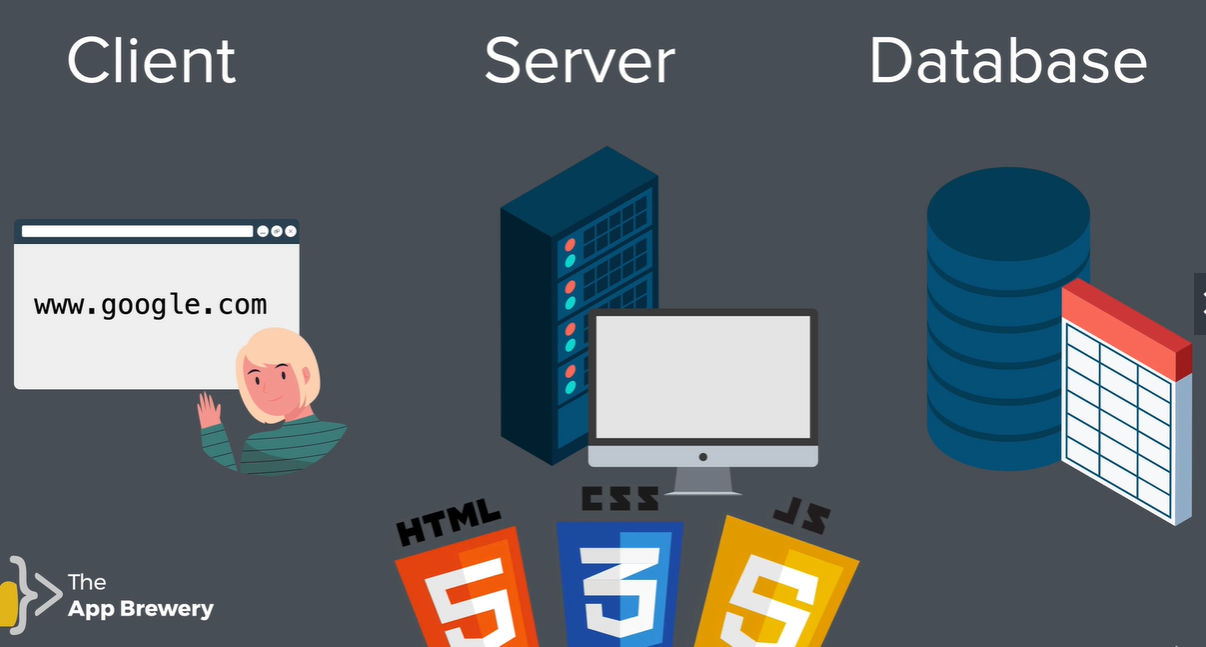


and a whole bunch more. The popular ones, if you look at job search

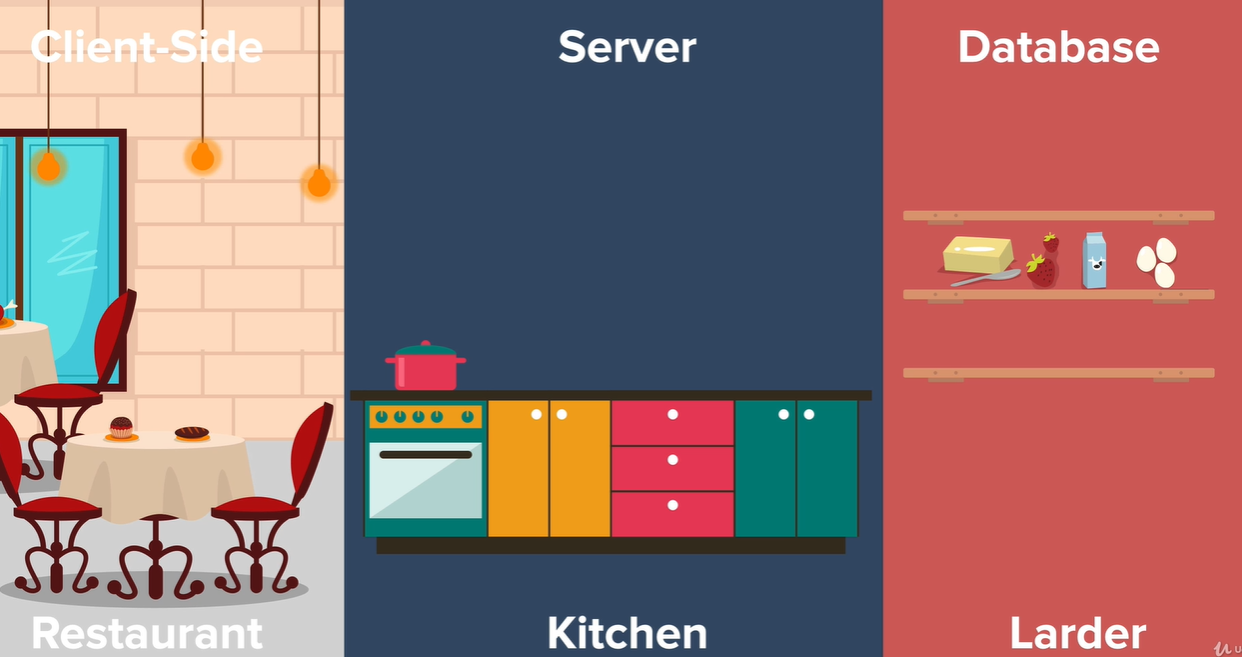
anyways are Flask and Django

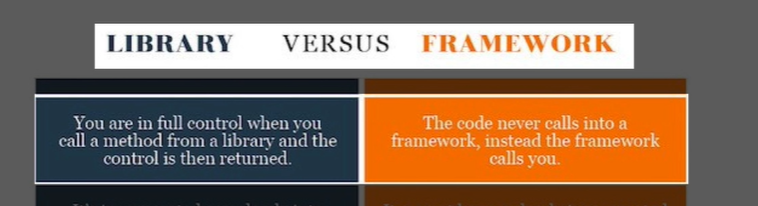
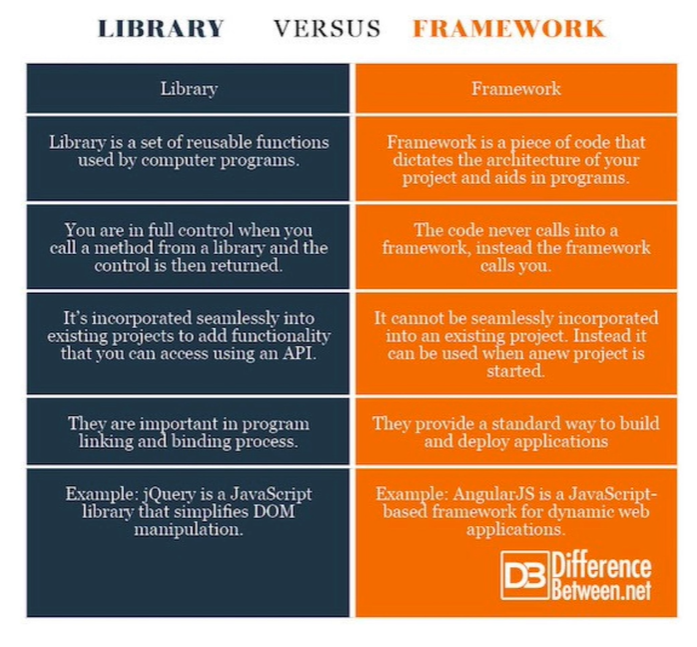


where Flask is better suited to beginners and also small projects, Django better for larger commercial projects .



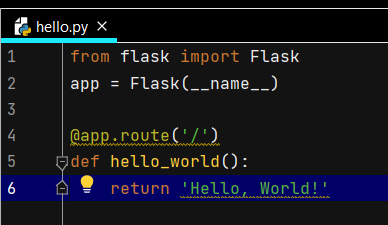
Restaurant analogy –



Flask Framework 

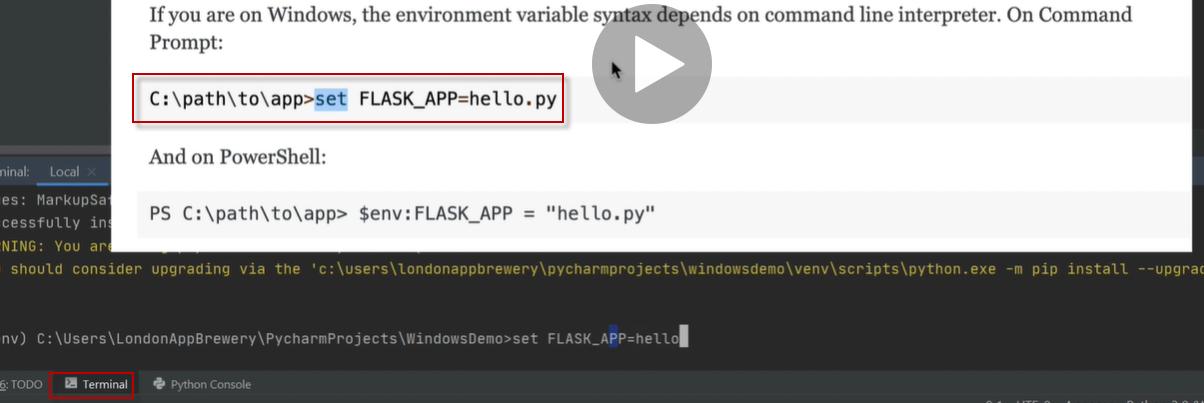
<https://pypi.org/project/Flask/>

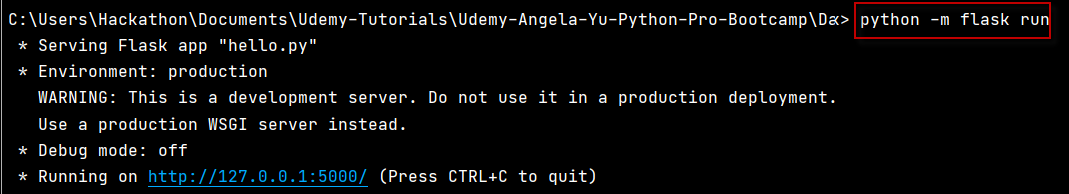
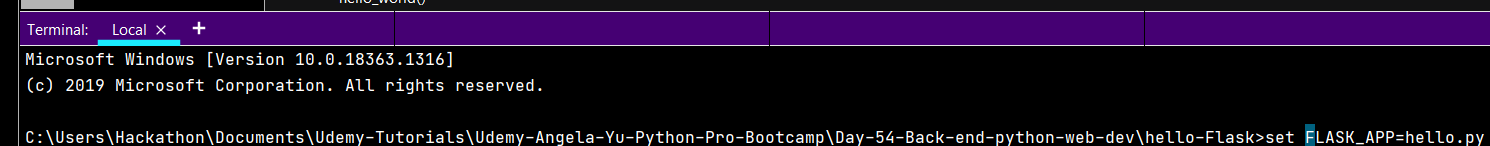
<https://flask.palletsprojects.com/en/1.1.x/quickstart/>



Need to export an environment variable . Tells the Flask Framework the name of the file that contains our server. Need to point to hello.py in order for Flask to recognize it and use it as the server

The name of the file in this example is hello.py

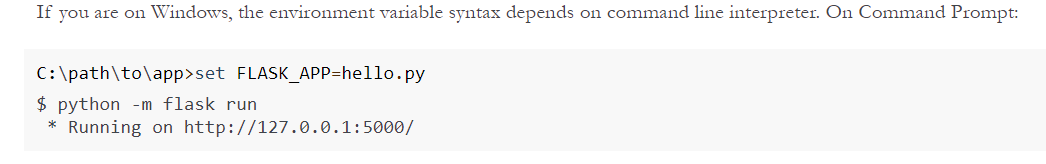


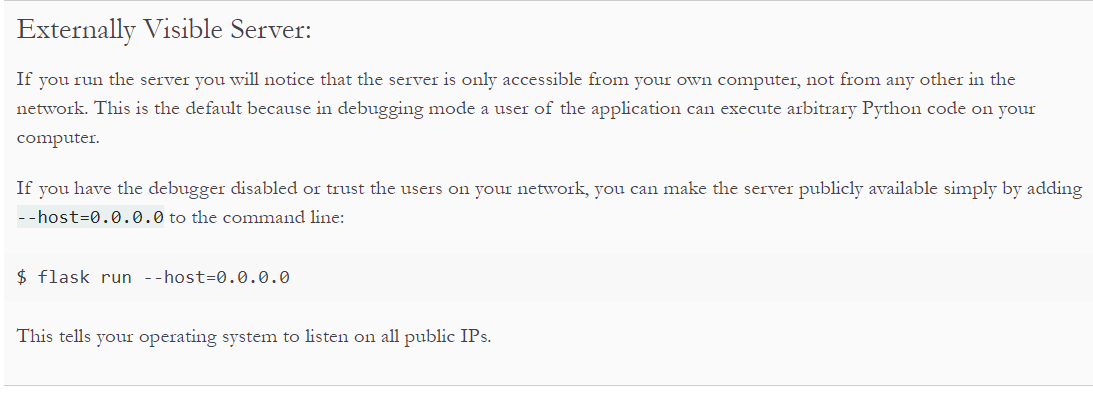


If you are on Windows, the environment variable syntax depends on command line interpreter. On Command Prompt:

C**:**\path\to\app>set FLASK\_APP=hello.py

python -m flask run





To stop the server CNTL c

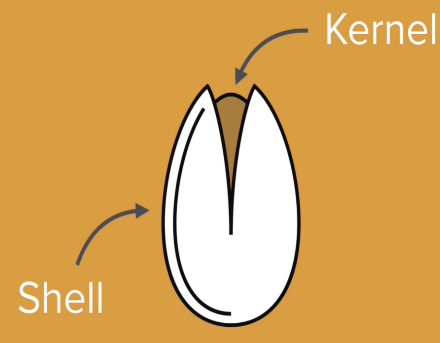
Running in Debug Mode will auto restart the server on code changes

$ set FLASK\_ENV=development

$ flask run

$ python -m flask run

Command line in the terminal , the command line , or the shell



Shell can be a GUI or on the command line - command line gives better control than the GUI

It can be faster than GUI

Windows default shell is the command prompt , I am using Bash

Print working directory pwd

List ls

Move around using cd

Create a new folder with mkdir

**To auto restart server with code changes run in debug mode**

Run app , then

Then in terminal

set FLASK\_APP=hello.py

set FLASK\_ENV=development

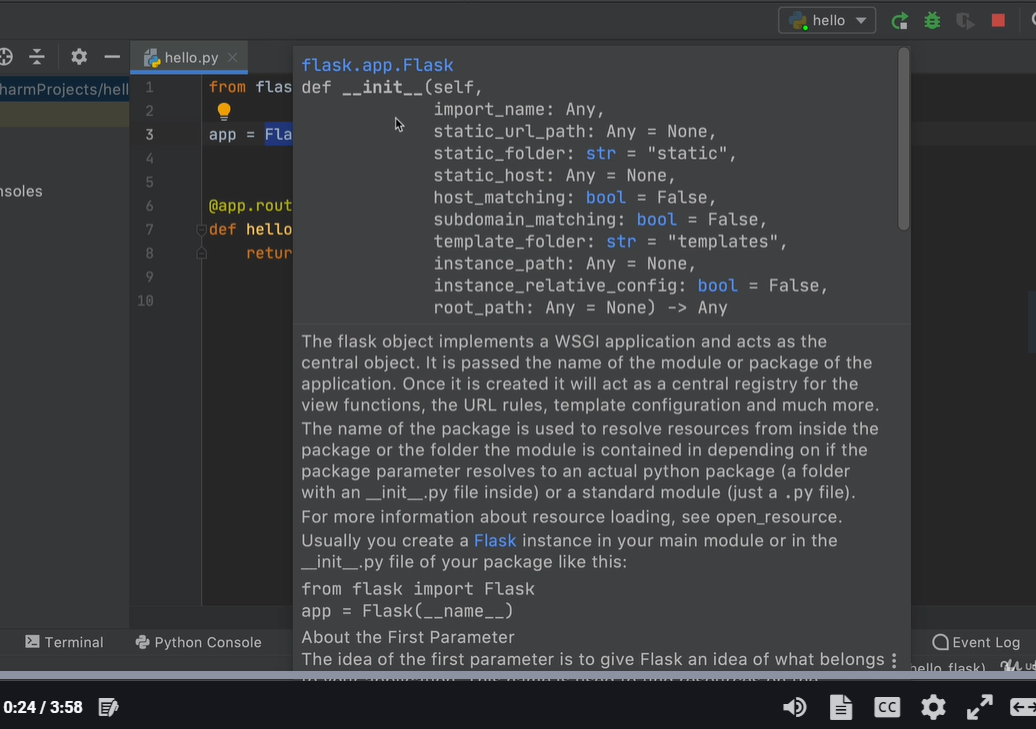
python -m flask run

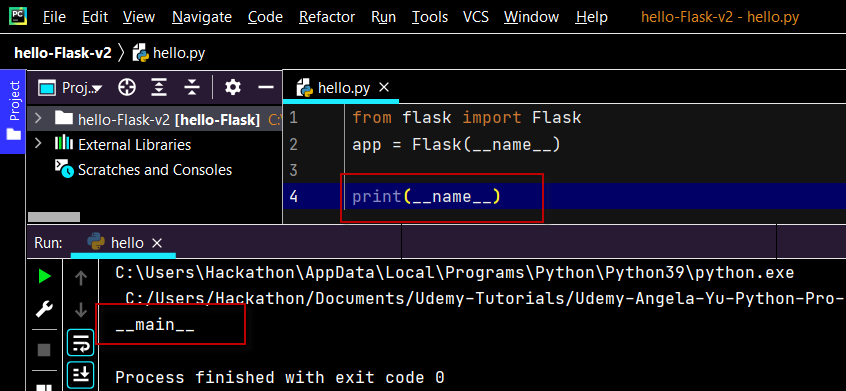
<https://docs.python.org/3/library/__main__.html>

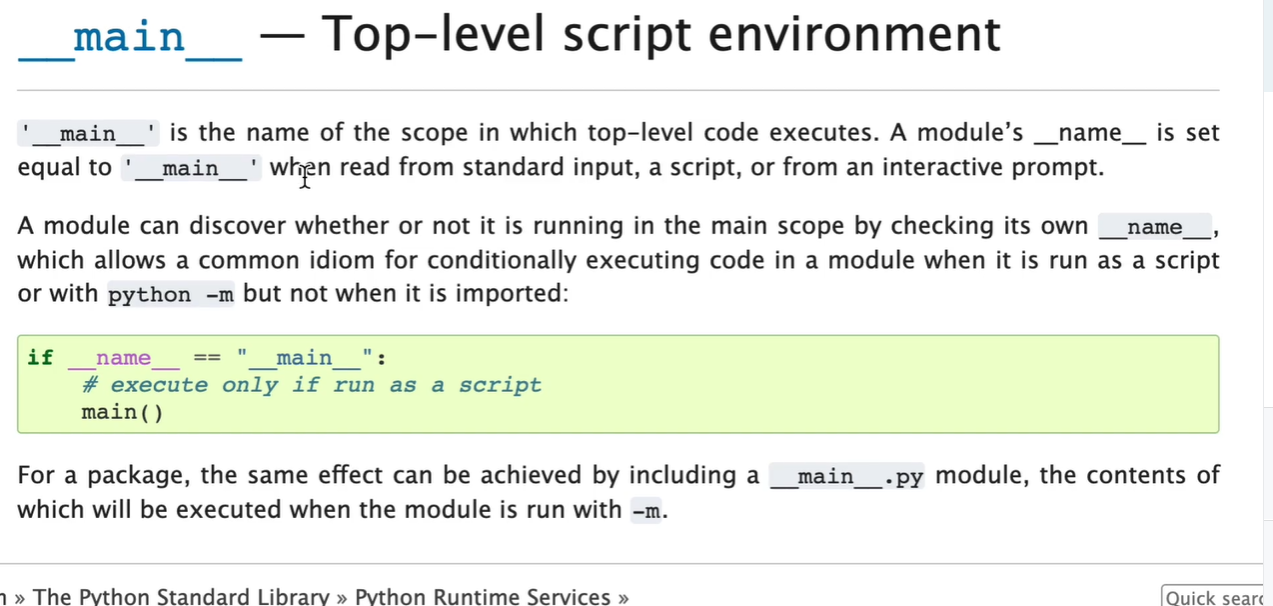
<https://docs.python.org/3/library/stdtypes.html?highlight=__name__#special-attributes>

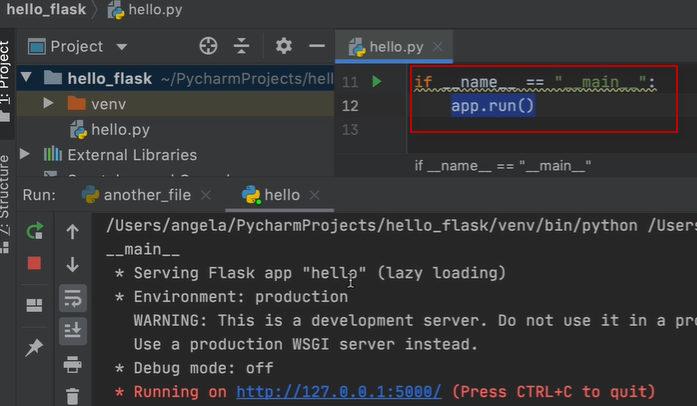
\_\_name\_\_ special attribute

One required import for the flask app









if \_\_name\_\_ is double equal to \_\_main\_\_ as a string,-- we're running the code from within this current file.We're running hello.py.

tap into our app and call the run method.

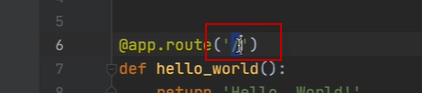
Now this app.run basically does the same thing as when we went into the terminal and we said flask run.

But with flask run, we have to provide the FLASK\_APP environment variable and we have to stop the code using control + c instead of using our normal run and stop.

With app.run, we can use our standard controls, simply hit run to run this hello.py, -it will start serving up our Flask app just as it did before, when we did flask run. Instead of using control + c to quit, simply use the stop to stop our Flask application.

**Python Decorator syntax**

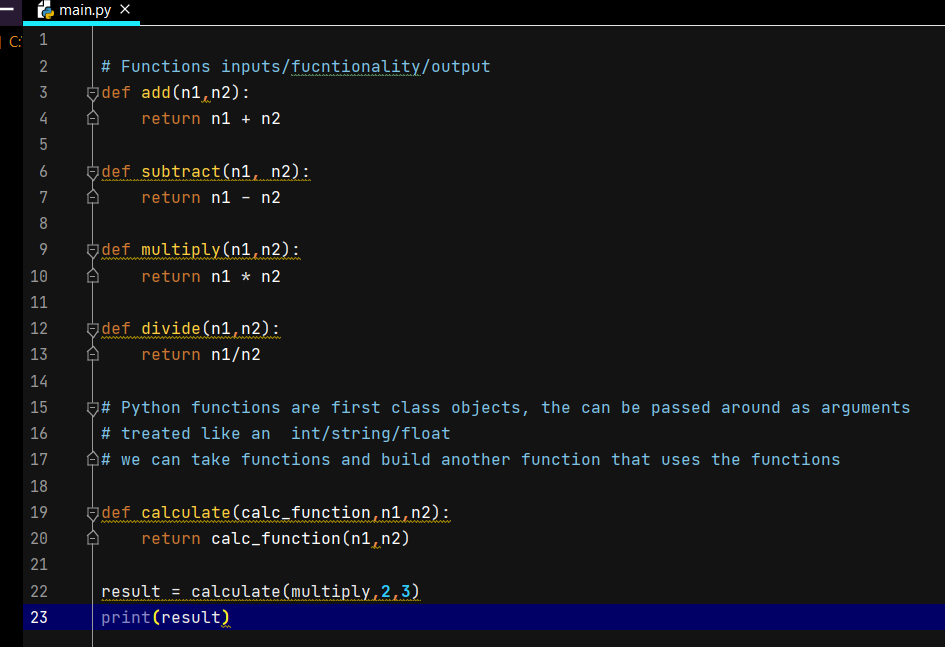
[**https://wiki.python.org/moin/PythonDecorators**](https://wiki.python.org/moin/PythonDecorators)



Go to the home page

A decorator function is a function gives additional functionality to an existing function

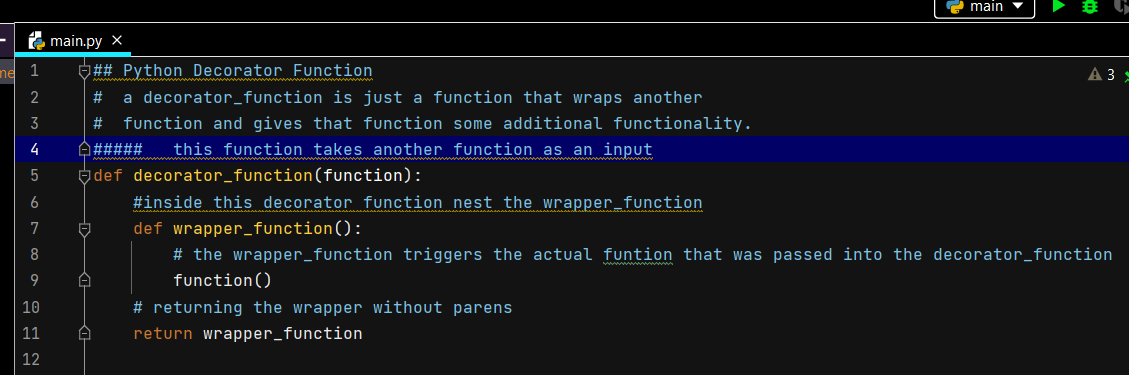
Python functions are first class objects

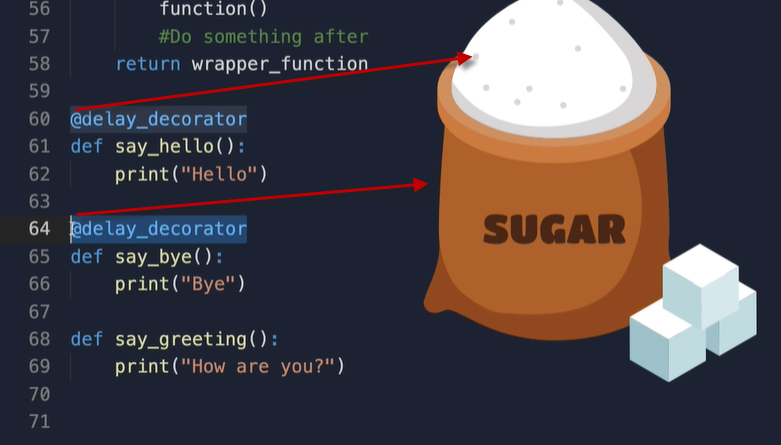


Python Tutor

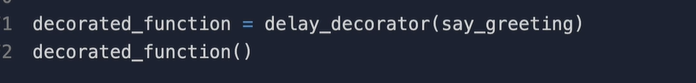
<http://www.pythontutor.com/visualize.html#mode=edit>

a decorator\_function is just a function that wraps another function and gives that function some additional functionality.





Without the @ code would look like this



Flask is decorating

