script environment

Script environment

Start of transcript. Skip to the end.

>> So, a nice thing about Python programs is that

they can call other Python programs.

In this section, we'll be discussing how you can change

Python environment variables to define the main program,

and how that would be used when running your own.

>> Let's look at running some

0:00 / 6:22 1.0x

Video

Download video file

Transcripts

Download SubRip (.srt) file

Download Text (.txt) file

Concepts

A Python script can be executed directly or imported as a module in another script. When running a script, a Python interpreter goes through a special setup procedure that defines some environment variables, one of those variables is the `__name__` variable, and it can distinguish between the two cases. If the script is being executed directly then `__name__` will contain the string `"__main__"`; otherwise, it will contain the name of the module. This distinction allows you to run parts of the code when the script is run directly; while executing other parts when the script is imported as a module.

Running a script directly

If a script (`main_script.py`) contains the following code:

```
print(__name__)
```

Running the script from a terminal window, will give you:

```
$ python main_script.py
__main__
```

Importing the script as a module

If the script (`main_script.py`) containing:

```
print(__name__)
```

Is imported into another script (`secondary_script.py`) that contains the following code:

```
import main_script
```

Running (`secondary_script.py`) will give you the name of the imported module (which is the name of the `main_script.py` file in this case)

```
$ python secondary_script.py
main_script
```

`main()` function

Generally, you test the value of `__name__` and execute a (`main()`) function if you are running the script directly. Otherwise, you do not run any function.

```
if __name__ == "__main__":
    main()
```

Examples

In the following example, `main_script.py` is run directly then imported as a module into `secondary_script.py`. You will see the content of `__name__` will change how the `main_script.py` code is executed.

Change working directory to `command_line`

Necessary so all generated files are saved in this directory, the cell will generate an error message if you are already in the `command_line` directory.

```
%cd command_line
```

`main_script.py`

The first line saves the Python code in the cell as `main_script.py` in the current working directory.

```
%%writefile main_script.py

# Start of Python code

# Will be called from another script
def func():
    print("Running func")

# Execute when running the script directly
def main():
    print("Running the main function")

if __name__ == "__main__":
    main()
```

Running `main_script.py` directly

The first line is necessary to run the rest of the lines as command lines (more on that later). For now, the `main_script.py` is being run directly

```
%%bash

python3 main_script.py
```

`secondary_script.py`

The first line saves the Python code in the cell as `secondary_script.py` in the current working directory.

The `secondary_script.py` imports `main_script` as a module and calls its `func()` function

```
%%writefile secondary_script.py

# Start of Python code
import main_script

# call func() from the main script
main_script.func()
```

Running `secondary_script.py` directly

The first line is necessary to run the rest of the lines as command lines (more on that later). For now, the `secondary_script.py` is being run directly.

```
%%bash

python3 secondary_script.py
```

Task 1

Script environment

```
# [ ] The following program asks the user for a circle radius then
# Modify the program so it only displays the information when exec
# the program should not display anything if it is imported as a m

from math import pi

def circle_area(r):
    return pi * (r ** 2)

def circle_circumference(r):
    return 2 * pi * r

radius = float(input("Enter radius: "))
print("Area =", circle_area(radius))
print("Circumference =", circle_circumference(radius))
```

Learn About Verified Certificates