**What does if \_\_name\_\_ == ”\_\_main\_\_” do in Python?**

**When and how a main method is executed in Python**



Photo by [Blake Connally](https://unsplash.com/@blakeconnally) on [unsplash.com](https://unsplash.com/photos/B3l0g6HLxr8)

If you are new to Python, you might have noticed that it is possible to run a Python script with or without a main method. And the notation used in Python to define one (i.e. if \_\_name\_\_ == ‘\_\_main\_\_') is definitely not self-explanatory especially for new comers.

In today’s tutorial we’ll explore and discuss the purpose of a main method and what to expect when you define one in your Python applications.

**What is the purpose of \_\_name\_\_ ?**

Before executing a program, the Python interpreter assigns the name of the python module into a special variable called \_\_name\_\_. Depending on whether you are executing the program through command line or importing the module into another module, the assignment for \_\_name\_\_ will vary.

If you invoke your module as a script, for instance

python my\_module.py

then Python Interpreter will automatically assign the string'\_\_main\_\_' to the special variable \_\_name\_\_. On the other hand, if your module is imported in another module

# Assume that this is another\_module.py  
import my\_module

then the string 'my\_module' will be assigned to \_\_name\_\_ .

**How does the main method work?**

Now let’s assume that we have the following module, that contains the following lines of code:

# first\_module.py  
print('Hello from first\_module.py')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print('Hello from main method of first\_module.py')

So in the module above, we have one print statement which is outside of the main method and one more print statement which is inside. The code under the main method, will only be executed if the module is invoked as a script from (e.g.) the command line, as shown below:

python first\_module.py  
Hello from first\_module.py  
Hello from main method of first\_module.py

Now, let’s say that instead of invoking module first\_module as a script, we want to import it in another module:

# second\_module.py  
import first\_module  
  
print('Hello from second\_module.py')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print('Hello from main method of second\_module.py')

And finally, we invoke second\_module as a script:

python second\_module.py  
Hello from first\_module.py  
Hello from second\_module.py  
Hello from main method of second\_module.py

Notice that the first output comes from module first\_module and specifically from the print statement which is outside the main method. Since we haven’t invoked first\_module as a script but instead we have imported it into second\_module the main method in first\_module will be simply ignored since if \_\_name\_\_ == ‘\_\_main\_\_' evaluates to False. Recall that from the above call, \_\_name\_\_ variable for second\_module has been assigned string '\_\_main\_\_' while the first\_module ‘s \_\_name\_\_ variable has been assigned the name of the module, i.e. ’first\_module’ .

Although everything under if \_\_name\_\_ == ‘\_\_main\_\_' is considered to be what we call a “main method”, it is a good practice to define one proper main method instead, which is called if the condition evaluates to True. For instance,

# my\_module.py  
def main():  
 """The main function of my Python Application"""  
 print('Hello World')  
  
if \_\_name\_\_ == '\_\_main\_\_':   
 main()

*Note: I would generally discourage you from having multiple main functions in a single Python application. I have used two different main methods just for the sake of the example.*

**Final Thoughts**

In this article, I have described how the main method gets executed in Python and under what conditions. When a module is invoked as a string, then Python interpreter will assign the string '\_\_main\_\_' to a special variable called \_\_name\_\_ and the code which is defined under the condition if \_\_name\_\_ == ‘\_\_main\_\_' will subsequently get executed. On the other hand, when a module is imported in another module, then the Python interpreter will assign the string with the name of that module to the special variable \_\_name\_\_ . This means that in such cases if \_\_name\_\_ == ‘\_\_main\_\_' will evaluate to False which means that only the code outside of this condition will get executed once imported.