07. Modules

October 2, 2025

1 7. Modules

In Python, a module is a file containing Python code, containing definitions of functions, classes, variables and executable statements. Modules allow you to organize code into reusable units and can be imported into other Python scripts or interactive sessions.

The name of the module (in our Python program) will be the name of the file. This is why the same naming conventions we described for variables also apply to module names.

1.1 7.1 Creating modules

A module is simply a Python file with a `.py` extension. You can define functions, classes, variables, or any other Python code within a module.

Here is an example of a module: my_module.py

1.1.1 Exercises 7.1

1. Place all the code you wrote for exercise 5.5 inside a module called `string_helper.py`.

1.2 7.2 Importing modules

To use functions, classes, or variables defined in a module, you need to import the module into your Python script. One option is to import the module and then use names inside the module prefixed by the module's name:

[3]:
```
import my_module
```

[4]:
```
my_module.get_words("A Python module can contain executable statements,␣
↪constants, functions and classes .")
```

[4]:
```
['a',
 'python',
 'module',
 'can',
 'contain',
 'executable',
 'statements',
 'constants',
 'functions',
```

```
 'and',
 'classes',
 '']
```

[3]:
```
help(my_module.get_words)
```

```
Help on function get_words in module my_module:

get_words(sentence)
    Returns a list of lowercase words contained in sentence
```

[4]:
```
help(my_module)
```

```
Help on module my_module:

NAME
    my_module - This module defines functions useful for string manipulation

FUNCTIONS
    get_words(sentence)
        Returns a list of lowercase words contained in sentence

DATA
    PUNCTUATION_MARKS = ',.!?;-*'

FILE
    /Users/iulia/PycharmProjects/python-beginner-course/docs/my_module.py
```

[5]:
```
my_module.PUNCTUATION_MARKS
```

[5]:
```
',.!?;-*'
```

Another option is to import specific items from the module:

[6]:
```
from my_module import PUNCTUATION_MARKS, get_words
print(PUNCTUATION_MARKS)
```

```
,.!?;-*
```

[7]:
```
get_words("In Python, a module is a file containing Python code.")
```

[7]:
```
['in',
 'python',
 'a',
 'module',
 'is',
```

```
 'a',
 'file',
 'containing',
 'python',
 'code']
```

Modules or specific names can also be imported under an alias, to avoid name conflicts, to provide a shorter or better name.

[8]:
```
from my_module import PUNCTUATION_MARKS as PUNCTUATION
print(PUNCTUATION)
```

```
,.!?;-*
```

Regardless of the import strategy (importing the entire module or specific names from it), the entire file is executed at import. Which means that, any executable statements the interpreter finds, it will execute them. Any statements you don't want to be executed at import must be placed under the following condition:

```
if __name__ == "__main__":
    # executable statements
```

The `__name__` variable is a special built-in variable that holds the name of the current module. When a Python script is executed, Python sets the value of `__name__` to `"__main__"` if the script is the main program being run. When a Python file is imported as a module into another script, Python sets the value of `__name__` to the name of the module.

For reasons of efficiency, a module is only loaded once per interpreter session. Meaning that, if the same module is imported from multiple locations in our running app, the import will be executed only once.

Also, while using the interactive interpreter, it may be the case that you need to reload an already imported module (because it has been edited and you need the latest version of it):

```
import mymodule
import importlib
importlib.reload(mymodule)
```

1.2.1 Exercises 7.2

1. Create a new module where you import `string_helper` and use the function. Try out different import versions.
2. Change `string_helper.py` so that the multiline string definition and the function call are not executed on import.

1.3 7.3 Module search path

When you import a module, Python searches for the module in a predefined list of directories known as the module search path. The module search path includes - the current directory - directories specified by the `PYTHONPATH` environment variable - the standard library directories.

You can view the module search path by accessing the `sys.path` list.

[9]:
```
import sys
sys.path
```

[9]:
```
['/opt/homebrew/Cellar/python@3.12/3.12.2/Frameworks/Python.framework/Versions/3
.12/lib/python312.zip',
 '/opt/homebrew/Cellar/python@3.12/3.12.2/Frameworks/Python.framework/Versions/3
.12/lib/python3.12',
 '/opt/homebrew/Cellar/python@3.12/3.12.2/Frameworks/Python.framework/Versions/3
.12/lib/python3.12/lib-dynload',
 '',
 '/Users/iulia/PycharmProjects/python-beginner-course/.venv/lib/python3.12/site-
packages']
```

1.4 7.4 Standard Library modules

Python comes with a standard library of modules that provide a wide range of functionality. You can import and use these modules in your Python scripts without installing additional packages.

Let's see some examples!

1.4.1 `os` module

The `os` module provides functions for interacting with the operating system, such as working with files and directories, managing processes, and handling file paths.

[10]:
```
import os
```

[11]:
```
os.getcwd()
```

[11]:
```
'/Users/iulia/PycharmProjects/python-beginner-course/docs'
```

1.4.2 `math` module

The `math` module provides mathematical functions and constants for performing mathematical operations, such as trigonometric functions, logarithmic functions, and constants like (pi) and e.

[12]:
```
import math
```

[13]:
```
math.sqrt(9)
```

[13]:
```
3.0
```

[14]:
```
math.pi
```

[14]:
```
3.141592653589793
```

1.4.3 `random` module

The `random` module provides functions for generating random numbers and performing random selections, such as generating random integers, shuffling sequences, and choosing random elements

from sequences.

[15]:
```
import random
```

[16]:
```
random.randint(1, 10)
```

[16]:
```
9
```

1.4.4 Exercises 7.4

1. Choose one of the following Python modules:
   - os
   - math
   - random
   - json
   - csv

   Try out at least one function from the module you chose.