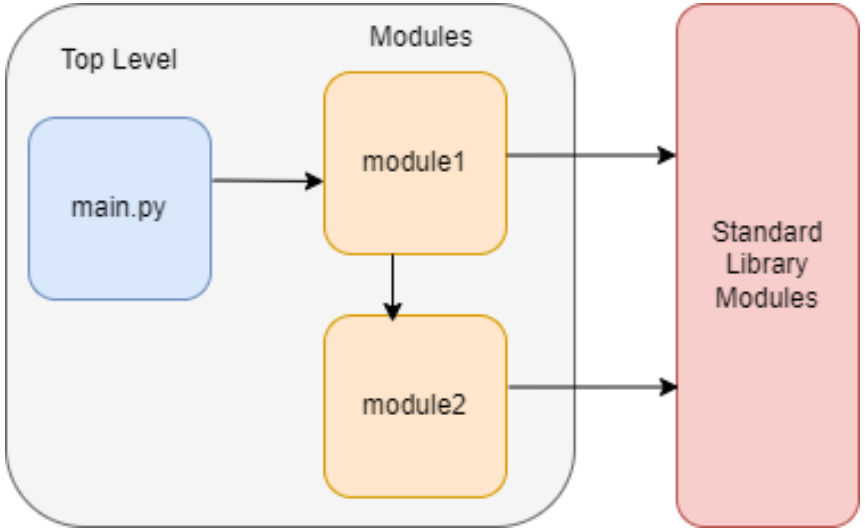
Modules and Package

Module Coding Basics, Module Packages

Modules

- Modules provide an easy way to organize components into a system by serving as self-contained packages of variables known as namespaces.
- All the names defined at the top level of a module file become attributes of the imported module object.
- Each file is a module, and modules import other modules to use the names they define.



Modules

- Modules are processed with two statements and one important function:
- $import$: Lets a client (importer) fetch a module as a whole
- $from$: Allows clients to fetch particular names from a module
- $imp.reload$: Provides a way to reload a module's code without stopping Python

How Imports Work

- Python module are really runtime operations that perform three distinct steps the first time a program imports a given file:

  - Find the module's file.
  - Compile it to byte code (if needed).
  - Run the module's code to build the objects it defines.

The module search path

- Python module search path is composed of the concatenation of four major components, some of which are preset for programmer and some which can be set.
  - The home directory of the program
  - PYTHONPATH directories(if set)
  - Standard library directories
  - The contents of any .pth files

Module Path

```
>>> import sys
>>> sys.path
['C:/MY DRIVE/Personal/Python program', 'C:\\Users\\Tumpa\\AppData\\Local\\Programs\\Python\\Python38\\Lib\\idlelib', 'C:\\Users\\Tumpa\\AppData\\Local\\Programs\\Python\\Python38\\python38.zip', 'C:\\Users\\Tumpa\\AppData\\Local\\Programs\\Python\\Python38\\DLLs', 'C:\\Users\\Tumpa\\AppData\\Local\\Programs\\Python\\Python38\\lib', 'C:\\Users\\Tumpa\\AppData\\Local\\Programs\\Python\\Python38', 'C:\\Users\\Tumpa\\AppData\\Roaming\\Python\\Python38\\site-packages', 'C:\\Users\\Tumpa\\AppData\\Local\\Programs\\Python\\Python38\\lib\\site-packages']
```

Module Example

module1.py - C:/MY DRIVE/Personal/Python program/module1.py (3.8.6rc1)

File Edit Format Run Options Window Help

```
def display(x):
    print('this is inside a function',x)
'''
import module1
module1.display('MCA')
'''
from module1 import display
display(10)
```

Module Packages

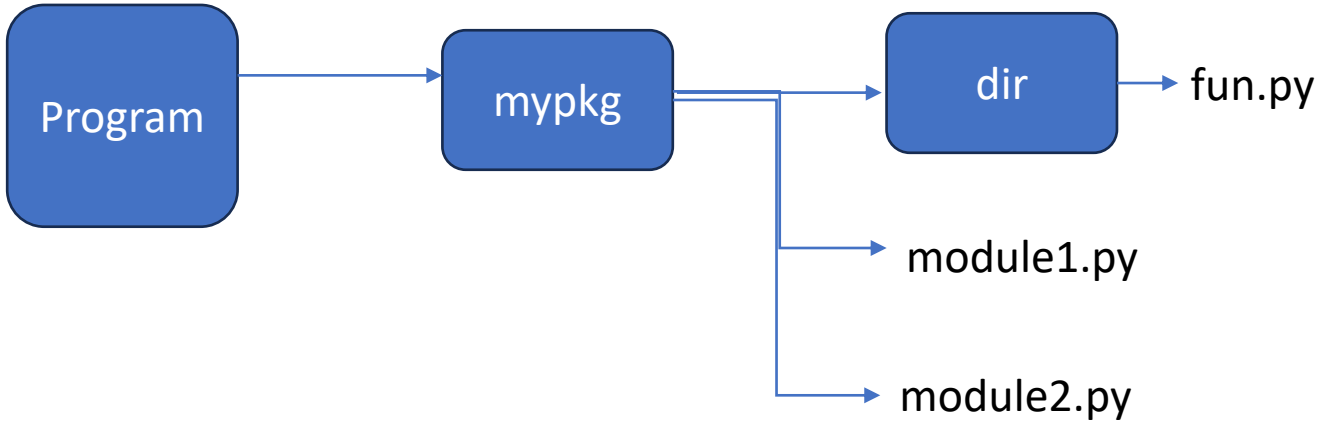
- A directory of Python code is said to be a package, so such imports are known as package imports. In effect, a package import turns a directory on your computer into another Python namespace

  - import dir1.dir2.mod
  - from dir1.dir2.mod import x

- each directory named within the path of a package import statement must contain a file named __init__.py, or your package imports will fail.

Example of a package

- Packages allow for a hierarchical structuring of the module namespace using dot notation.
- use hierarchical file structure of the operating system.



Example

module1.py - C:/MY DRIVE/Personal/Python program/mypkg/module1.py (3.8.6rc1)

File Edit Format Run Options Window Help

```
def os():
    print('this is my OS subject')
def ds():
    print('this is my ds subject')
```

module2.py - C:/MY DRIVE/Personal/Python program/mypkg/module2.py (3.8.6rc1)

File Edit Format Run Options Window Help

```
def sunday():
    print('Weekend')
def monday():
    print('first day of the week')
```

fun.py - C:\MY DRIVE\Personal\Python program\mypkg\dir1\fun.py (3.8.6rc1)

File Edit Format Run Options Window Help

```
def fun():
    print('This is directory withing a package')
```

mod1.py - C:/MY DRIVE/Personal/Python program/mod1.py (3.8.6rc1)

File Edit Format Run Options Window Help

```
from mypkg import module1
from mypkg.dir1 import fun
module1.os()
fun.fun()
```