

## Modules and Packages

Modules:- A group of functions, variables and classes saved a file, which is nothing but a module and it's in build.

ex:- math, os, sys, functools, unittest etc.

Every:- Python file show as a module.

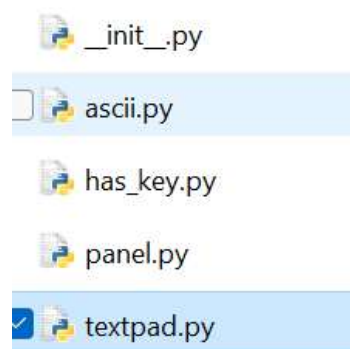
```
class Template:
    """A string class for supporting $-substitutio
    delimiter = '$'

    idpattern = r'(?a:[_a-z][_a-z0-9]*)'
    braceidpattern = None
    flags = _re.IGNORECASE

    def __init_subclass__(cls):
        super().__init_subclass__()
        if 'pattern' in cls.__dict__:
            pattern = cls.pattern
        else:
```

Packages :- It is an encapsulation mechanism to group related modules in a single unit, it's a package of module. It's look like a folder and it's contain several python files(.py) and \_\_init\_\_.py(imp)

Ex:- pandas, numpy, pytest, tensorflow etc



Using **import** keyword we call python module

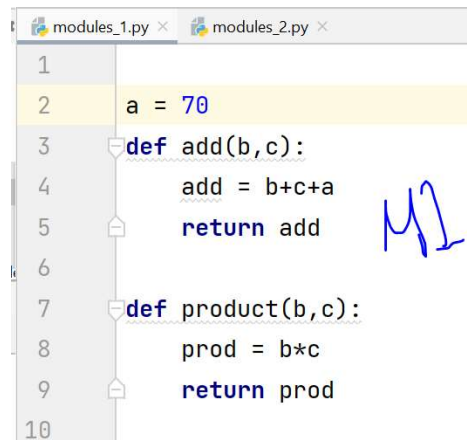
```
import math
import opcode
import os
import datetime
import sys
import zipfile
import json
import xml
```

```
print(dir(math))
```

```
>>> import math
>>> dir(math)
['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos', 'acosh', 'asin', 'asinh', 'atan',
'atan2', 'atanh', 'ceil', 'comb', 'copysign', 'cos', 'cosh', 'degrees', 'dist', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs',
'factorial', 'floor', 'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose', 'isfinite', 'isinf', 'isnan',
'isqrt', 'lcm', 'ldexp', 'lgamma', 'log', 'log10', 'log1p', 'log2', 'modf', 'nan', 'nextafter', 'perm', 'pi', 'pow',
'prod', 'radians', 'remainder', 'sin', 'sinh', 'sqrt', 'tan', 'tanh', 'tau', 'trunc', 'ulp']
>>> math.pi
3.141592653589793
>>> math.pow(3,2)
9.0
```

**How to create user define module.:-**

## Create a python file, and create class, functions and variables



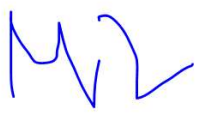
```
1
2 a = 70
3 def add(b,c):
4     add = b+c+a
5     return add
6
7 def product(b,c):
8     prod = b*c
9     return prod
10
```

**Note:-** If we want call this python file as a module, create another python file and using import keyword, import the module .

Ex:- 1st file :- modules\_1.py  
2nd file:- module\_2.py

Import modules\_1.py

```
modules_1.py x modules_2.py x
1 import modules_1
2 print(dir(modules_1))
3 print(modules_1.add(40,30))
4 print(modules_1.product(7,8))
```



**Note:-** Whenever we are using a module in our program . For that module complied file will be generated and stored in the disk permanently .

Class-2

## Renaming a module at the of import(Module aliasing)

Using as keyword we will alias module name.

```
import math as m
print(dir(m))
```

Here math is a original module name and m is a alias name  
We can access member(methods) by using alias name m.

## from.....import

We can import particular member of module by using from...import.  
The main advantage of this is we can access member directly without using module name, and also we reduce the storage.

```
from math import factorial,pow,pi
print(factorial(5))
print(pow(2,3))
print(pi)
```

**Note:-** We can import all member of module using \*.

```
from math import *  
print(factorial(12))
```

## Various Possibility of import.

**Import** module name

**Import** module1,module2,module3

**Import** module **as** m

**Import** module1 **as** m1,module2 **as** m2,module3 **as** m3

**From** module **import** member

**From** module **import** member1,member2,member3

**From** module **import** \*

## Important Basic Modules.

```
import math  
import opcode  
import os  
import datetime  
import sys  
import zipfile  
import json  
import xml
```

```
print(dir(math))
```

# Math , os , sys, datetime (Very very important)

Math :-[https://www.w3schools.com/python/module\\_math.asp](https://www.w3schools.com/python/module_math.asp)

**Note:-** Last we will discuss Json

## The Special Variable (\_\_name\_\_)

Fro every Python program a special variable `__name__` will be added internally. This Variable stores information regarding whether the program is executed as an individual program as a module.

If the program executed as an individual program then the value a variable is `__main__`.

```
>>> import datetime
>>> dir(datetime)
['MAXYEAR', 'MINYEAR', '__all__', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package__', '__spec__', 'date', 'datetime', 'datetime_CAPI', 'sys', 'time', 'timedelta', 'timezone', 'tzinfo']
>>>
```

```
def f1():
    if __name__ == 'main':
        print('The code excutaed as a program')
    else:
        print('The code executed as a module from some other program')

print(f1())
```