Modules and Packages

ماژول‌ها و بسته‌ها

در این بخش به طور خلاصه در مورد ۱- نوشتن یک ماژول ساده و اضافه کردن آن به کد پایتون، ۲- اجرای کد پایتون در یک سلول ژوپیتر و ۳- نشان دادن این که می‌توانیم یک سری ورودی را به فایل اسکریپت پایتون بدهیم بحث خواهیم. کرد.

اول ببنیم که یک ماژول چی هست؟

ماژول‌ها در حقیقت یک فایل هستند که درون آن‌ها توابع، تعاریف و دستورات مربوط به یک کاربرد خاص قرار دارد. و ما با اضافه کردن اون ماژول به برنامه خود، توانایی‌های جدید به کدمان اضافه می‌کنیم!

ما از ماژول‌ها به این دلیل استفاده می‌کنیم که یک مجموعه بزرگ از عملکردها را خلاصه و در دسترس و مدیریت پذیر کنیم. یعنی فرض کنید تعداد زیادی تابع را که عملکردی مرتبط دارند را به صورت منظم دسته بندی و استفاده کنید. یک فایل متنی باز کنید و درون آن دستوراتی را بنویسید. پسوند فایل را به `py.` تغییر دهید، مثلا فایل `module1.py` را در نظر بگیرید. در ژوپیتر نوت‌بوک برای این که بتوانید یک سلول را درون یک فایل متنی بنویسید، کد زیر را ابتدای سلول قرار دهید:

```
%%writefile module1.py
```

In [1]:
```
%%writefile module1.py
# Python Module example
def add(a, b):
    """This program adds two
    numbers and return the result"""

    result = a + b
    return result
```

```
Writing module1.py
```

In [3]:
```
add(2,3)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-9cd3bc5fb6d1> in <module>()
----> 1 add(2,3)

NameError: name 'add' is not defined
```

حالا یه ماژول داریم، چطور از آن استفاده کنیم؟

ما می‌توانید آنچه را که درون این ماژول نوشته شده است را با دستور `import` به برنامه اضافه کنیم.

In [4]:
```
import module1
```

In [5]:
```
add(2,3)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-9cd3bc5fb6d1> in <module>()
----> 1 add(2,3)

NameError: name 'add' is not defined
```

In [6]:
```
module1.add(2,3)
```

Out[6]: 5

لازم هست وقتی می‌خواهیم از تابع add استفاده کنیم، حتما بایداسم ماژول را هم بنویسیم.

برای این که همه چیز تحت کنترل باشد، بهتر است که اینطور باشد، در پایتون ماژول‌های خیلی زیادی هست که شاید این تابعی که در این ماژول تعریف کرده‌ایم، در ماژول‌های دیگر هم بکار رفته باشد، در این صورت برای جلوگیری از تداخل و سردرگمی این کار بهترین کار است.

```
module_name.function_name(args)
```

یک راه دیگر هم هست که خودتان را راحت کنید و فقط نام تابع را بنویسید، ولی خب حواستان باشد که ممکن است توابعی که از این ماژول وارد برنامه می‌شوند، جایگزین توابع هم‌نامی شوند که پیش از این در برنامه بودند.

```
from module_name import function_name
from module_name import *
```

به نظر میاد که نوشتن اسم تابع یا ماژول سخت یا حوصله سر بر باشد! دستور import فکر این را هم کرده است!

```
# rename the module:
import math as mt
from matplotlib import pyplot as plt
```

بگذارید ماژول خود را تغییر دهیم و به آن چند متغیر و تابع اضافه کنیم.

In [7]:
```
%%writefile module1.py
# Python Module example
def add(a, b):
    """This program adds two
    numbers and return the result"""

    result = a + b
    return result


def subtract(a, b):
    """This program subtracts two
    numbers and return the results"""

    result = a - b
    return result


the_variable = 1
variable1 = 2
```

```
Overwriting module1.py
```

In [8]:
```
from module1 import add
```

In [9]:
```
add(2,3)
```

Out[9]: 5

```
In [10]: subtract(3,2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-10-631ae9686a8c> in <module>()
----> 1 subtract(3,2)

NameError: name 'subtract' is not defined
```

```
In [11]: module1.subtract(3,2)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-11-dc5f08ed04f4> in <module>()
----> 1 module1.subtract(3,2)

AttributeError: module 'module1' has no attribute 'subtract'
```

ما از این ماژول فقط یک تابع را وارد برنامه کردیم و فقط همان در دسترس ما است.

```
In [12]: import module1
```

```
In [13]: module1.subtract(3,2)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-13-dc5f08ed04f4> in <module>()
----> 1 module1.subtract(3,2)

AttributeError: module 'module1' has no attribute 'subtract'
```

الان چه اتفاقی افتاد به نظر تان؟

ما یک بار قبلا ماژول یک را به برنامه اضافه کرده بودیم. و در ادامه ماژول را تغییر دادیم. و دوباره آن را وارد کردیم، اما دستورات جدید آن وارد برنامه نشدند! علتش این است که مفسر پایتون ماژول‌ها را یک بار بیشتر به برنامه اضافه نمی‌کند. و این صرفا برای سرعت بخشیدن به کار است. حال اگر ماژول در حین یک نشست تغییر کرده باشد، بنابراین نیاز است که آن را بازخوانی کرد. این کار با استفاده از دستور `reload` از ماژول `imp` انجام می‌شود.

```
In [15]: import imp
```

```
In [16]: imp.reload(module=module1)
```

```
Out[16]: <module 'module1' from '/home/anncourse/Desktop/python_class/Complete-Python
-3-Bootcamp/06-Modules and Packages/module1.py'>
```

```
In [17]: module1.subtract(5,2)
```

```
Out[17]: 3
```

```
In [18]: module1.the_variable
```

```
Out[18]: 1
```

حال بیایید در مورد بسته‌ها صحبت کنیم.

ممکن است در یک کاربرد خاص، مثلا حل مسائل ریاضی، نیاز به ماژول‌های مختلف داشته باشیم. در هر ماژول ابزارهای مختلفی هستند، مثلا در یک ماژول روابط مربوط به ماتریس‌ها و عملیات ماتریسی. در ماژول دیگر الگوریتم‌های ریاضی که از ماتریس‌ها استفاده می‌کنند. اما مسأله‌ای که وجود دارد این است که این ماژول‌ها نیز به هم مرتبط هستند. بجای این که هر کدام آن‌ها جداگانه یک گوشه روی رایانه ذخیره شوند، در ساختار آبشاری منظمی از پوشه‌ها قرار می‌گیرند که به آن بسته یا `package` می‌گوییم.

یک بسته خیلی بزرگ مانند `numpy` می‌تواند شامل هزاران ماژول شود.

بسته‌هاو ماژول‌ها را از کجا پیدا کنیم؟

یکسری از ماژول‌ها به صورت از پیش آماده در همراه نصب پایتون موجود هستند:

- sys
- os
- random
- ...

را دیگر این است که در اینترنت بگردید و آنچه را که نیاز دارید را پیدا کنید. اما پیش‌تر از آن با جای معتبرتری به نام مخزن `pypi` آشنا شدیم. یکسری از بسته‌های مورد نیاز زیر را نصب کنید.

```
pip install spyder
pip install numpy
pip install scipy
pip install sklearn
pip install matplotlib
```

```
In [19]: !pip install --user matplotlib
```

```
Requirement already satisfied: matplotlib in /usr/lib/python2.7/dist-packages
(2.2.3)
```

در ادامه به طور مختصر با برخی از این بسته‌ها آشنا خواهیم شد...