



## 4.2.2 模块、包和库

### ■ 模块 (Module)

- 模块是一个python文件 (.py) , 拥有**多个**功能相近的**函数**或**类**。
- 便于**代码复用**, 提高**编程效率**, 提高了代码的**可维护性**。
- 避免函数名和变量名冲突。

### ■ 包 (Package)

- 为了**避免模块名冲突**, Python引入**按目录来组织模块**的方法。
- 一个包对应一个**文件夹**, 将**功能相近**的模块 (Python文件), 放在同一个文件夹下。
- 在作为包的文件夹下有一个**\_\_init\_\_.py**文件。
- **子包**: 子目录中也有**\_\_init\_\_.py**文件。

### ■ 库 (Library): 具有相关功能的**模块或包的集合**。



### 例：包的结构

```
package_a
├── __init__.py
├── module_a1.py
├── module_a2.py
└── subpack_ab
    ├── __init__.py
    ├── module_ab_1.py
    └── module_ab_2.py
```

这个包对应文件夹package\_a，其中有一个\_\_init\_\_文件，和两个模块，还有一个子包；子包中也有2个模块。



### □ 导入模块/包/库

#### ■ 导入整个包

```
import 名称 as 别名
```

PEP8 Python  
编码规范推荐

```
import numpy as np  
np.random.random()
```

#### ■ 导入包中指定的模块或子包

```
from 模块名 import 函数名 as 函数别名
```

```
from numpy import *  
random.random()
```



### □ 导入语句的作用域：

- 在程序顶部导入模块，作用域是全局的。
- 在函数的内部导入语句，作用域就是局部的。

### 建议导入的顺序

python 标准库/模块  
python 第三方库/模块  
自定义模块



### □ 使用模块/包/库中的函数和变量

模块/包/库名.函数名(参数列表)

模块/包/库名.变量名

```
>>>import math                # 导入math模块
>>>math.pow(2,3)              # 计算2的3次幂
8.0

>>>from math import pow, sqrt  # 从math模块中只导入pow和sqrt函数
>>> pow(2,3)                  # 计算2的3次幂
8.0
>>>sqrt(16)                   # 计算16的开平方
4.0

>>>from math import sqrt as s  # 从math模块中导入sqrt函数并重命名为s
>>>s(16)
4.0
```



### □ 自定义模块

- 创建自定义模块：将常用函数的定义放在一个.py文件中。

mymodule.py

```
def 函数1():          #定义函数1
    函数体1

def 函数2():          #定义函数2
    函数体2
```

- 当函数较多时，可以**按照功能**将它们放在**不同的模块**中。
- 一个应用程序中可以定义多个模块。



### □ 创建自定义模块

mymodule.py

```
def print_str(str):           # 打印字符串
    print(str)

def sum(a,b):                 # 求和
    return a+b
```

### □ 调用自定义模块

```
>>>import mymodule as mm      # 导入mymodule模块
>>>mm.print_str("Python")     #调用打印函数
Python
>>>mm.sum(2,3)                #调用求和函数
5
```





### ■ Python标准库中的模块

- sys模块：提供有关Python运行环境的变量和函数。

#### sys模块中的常用变量

变 量	功能
sys.platform	获取当前操作系统
sys.path	获取指定模块搜索路径
sys.argv	获取当前正在执行的命令行参数的参数列表



- `sys.platform`: 获取当前操作系统。

```
>>>import sys
>>>sys.platform
linux2
```

```
>>>import sys
>>>sys.platform
win32
```

- `sys.path`: 获取指定模块搜索路径。

```
>>> import sys
>>> sys.path
['',
'F:\\python3.6\\Lib\\idlelib',
'F:\\python3.6\\python36.zip',
'F:\\python3.6\\DLLs',
'F:\\python3.6\\lib',
'F:\\python3.6',
'F:\\python3.6\\lib\\site-packages']
```



- `sys.append()`: 添加指定模块搜索路径。

```
sys.path.append('路径')
```

```
>>>sys.path.append('F:\\myProgame')
>>>sys.path
['',
 'F:\\python3.6\\Lib\\idlelib',
 'F:\\python3.6\\python36.zip',
 'F:\\python3.6\\DLLs',
 'F:\\python3.6\\lib',
 'F:\\python3.6',
 'F:\\python3.6\\lib\\site-packages',
 'F:\\myProgame']
```

在退出Python环境后，使用`sys.path.append()`函数添加的路径会**自动消失**。



### sys模块中的常用函数

函 数	功 能
<code>sys.exit(n)</code>	退出应用程序。 n=0, 正常退出; n=1, 异常退出。
<code>sys.getdefaultencoding()</code>	获取系统当前编码。
<code>sys.setdefaultencoding()</code>	设置系统默认编码
<code>sys.getfilesystemencoding()</code>	获取文件系统使用编码方式, Windows下返回'mbcs', mac下返回'utf-8'.



□ platform模块：获取操作系统的详细信息和与Python有关的信息。

函 数	功 能
platform.platform()	获取操作系统名称及版本号信息
platform.system()	获取操作系统类型
platform.version()	获取操作系统的版本信息
platform.processor()	获取计算机的处理器信息
platform.python_build()	获取Python的版本信息，包括Python的主版本、编译版本号和编译时间等信息



□ math模块：提供了常用的数学运算。

变量和函数	功能
math.e	返回自然对数e的值
math.pi	返回 $\pi$ 的值
math.exp(x)	返回e的x次幂
math.fabs(x)	返回x的绝对值
math.ceil(x)	返回大于等于x的最小整数
math.floor(x)	返回小于等于x的最大整数
math.log(x,a)	返回 $\log_a x$ ,如果不指定参数a, 则默认使用e
math.log10(x)	返回 $\log_{10} x$
math.pow(x,y)	返回x的y次幂
math.sqrt(x)	返回x的开平方



□ random模块：生成随机数。

函 数	功 能
random.random()	生成一个0到1的随机浮点数
random.uniform(a, b)	生成一个指定范围内的随机浮点数。其中a是下限，b是上限
random.randint(a, b)	生成一个指定范围内的随机整数。a是下限，b是上限
random.choice(seq)	从序列中随机获取一个元素。参数seq表示一个有序类型，可以是一个列表、元组或者字符串
random.shuffle(x)	将一个列表x中的元素打乱



### □ 小数和分数处理模块

- decimal模块：表示和处理小数。
- fractions模块：表示和处理分数。

### □ 时间处理模块

- time
- datetime
- calendar

