使用模块

阅读: 1094655

Python本身就内置了很多非常有用的模块，只要安装完毕，这些模块就可以立刻使用。

我们以内建的`sys`模块为例，编写一个`hello`的模块：

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

' a test module '

__author__ = 'Michael Liao'

import sys

def test():
    args = sys.argv
    if len(args)==1:
        print('Hello, world!')
    elif len(args)==2:
        print('Hello, %s!' % args[1])
    else:
        print('Too many arguments!')

if __name__=='__main__':
    test()
```

第1行和第2行是标准注释，第1行注释可以让这个`hello.py`文件直接在Unix/Linux/Mac上运行，第2行注释表示.py文件本身使用标准UTF-8编码；

第4行是一个字符串，表示模块的文档注释，任何模块代码的第一个字符串都被视为模块的文档注释；

第6行使用`__author__`变量把作者写进去，这样当你公开源代码后别人就可以瞻仰你的大名；

以上就是Python模块的标准文件模板，当然也可以全部删掉不写，但是，按标准办事肯定没错。

后面开始就是真正的代码部分。

你可能注意到了，使用`sys`模块的第一步，就是导入该模块：

```
import sys
```

导入`sys`模块后，我们就有了变量`sys`指向该模块，利用`sys`这个变量，就可以访问`sys`模块的所有功能。

`sys`模块有一个`argv`变量，用list存储了命令行的所有参数。`argv`至少有一个元素，因为第一个参数永远是该.py文件的名称，例如：

运行`python3 hello.py`获得的`sys.argv`就是`['hello.py']`；

运行`python3 hello.py Michael`获得的`sys.argv`就是`['hello.py', 'Michael]`。

最后，注意到这两行代码：

```
if __name__=='__main__':
    test()
```

当我们在命令行运行`hello`模块文件时，Python解释器把一个特殊变量`__name__`置为`__main__`，而如果在其他地方导入该`hello`模块时，`if`判断将失败，因此，这种`if`测试可以让一个模块通过命令行运行时执行一些额外的代码，最常见的就是运行测试。

我们可以用命令行运行`hello.py`看看效果：

```
$ python3 hello.py
Hello, world!
$ python hello.py Michael
Hello, Michael!
```

如果启动Python交互环境，再导入`hello`模块：

```
$ python3
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 23 2015, 02:52:03)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import hello
>>>
```

导入时，没有打印`Hello, word!`，因为没有执行`test()`函数。

调用`hello.test()`时，才能打印出`Hello, word!`：

```
>>> hello.test()
Hello, world!
```

作用域

在一个模块中，我们可能会定义很多函数和变量，但有的函数和变量我们希望给别人使用，有的函数和变量我们希望仅仅在模块内部使用。在Python中，是通过`_`前缀来实现的。

正常的函数和变量名是公开的（public），可以被直接引用，比如：`abc`，`x123`，`PI`等；

类似`__xxx__`这样的变量是特殊变量，可以被直接引用，但是有特殊用途，比如上面的`__author__`，`__name__`就是特殊变量，`hello`模块定义的文档注释也可以用特殊变量`__doc__`访问，我们自己的变量一般不要用这种变量名；

类似`_xxx`和`__xxx`这样的函数或变量就是非公开的（private），不应该被直接引用，比如`_abc`，`__abc`等；

之所以我们说，private函数和变量“不应该”被直接引用，而不是“不能”被直接引用，是因为Python并没有一种方法可以完全限制访问private函数或变量，但是，从编程习惯上不应该引用private函数或变量。

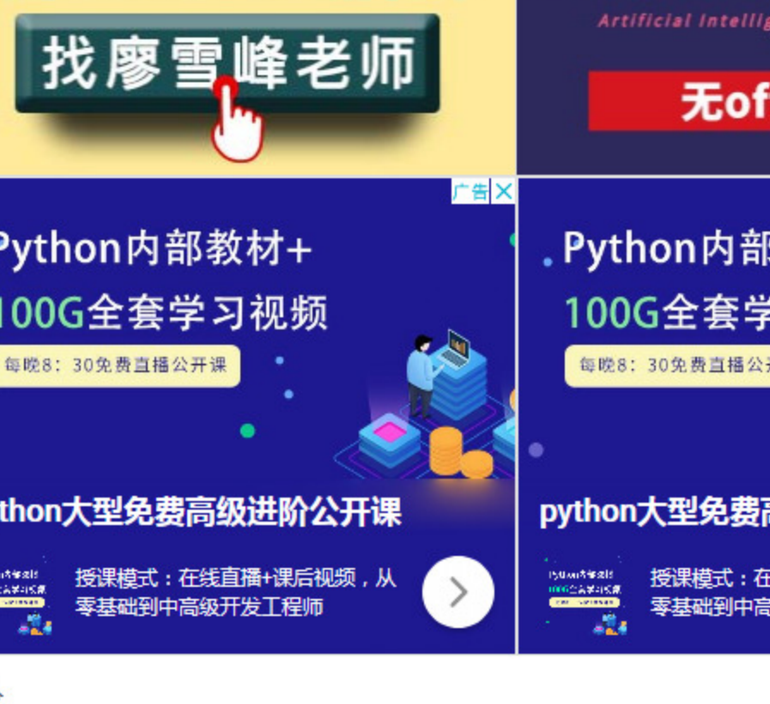
private函数或变量不应该被别人引用，那它们有什么用呢？请看例子：

```
def _private_1(name):
    return 'Hello, %s' % name

def _private_2(name):
    return 'Hi, %s' % name

def greeting(name):
    if len(name) > 3:
        return _private_1(name)
    else:
        return _private_2(name)
```

我们在模块里公开`greeting()`函数，而把内部逻辑用private函数隐藏起来了，这样，调用`greeting()`函数不用关心内部的private函数细节，这也是一种非常有用的代码封装和抽象的方法，即：

外部不需要引用的函数全部定义成private，只有外部需要引用的函数才定义为public。

读后有收获可以请作者喝咖啡，读后有疑问请加群讨论：



还可以分享给朋友：

分享到微博

上一页 下一页

评论

还有一种导入方法
金钟铉 created at 2 days ago, Last updated at 2 days ago

```
from 模块名 import xxx
```

全部讨论 回复

关于《if __name__ == '__main__':解释》的补充
ywjco_567 created at 3 days ago, Last updated at 3 days ago

官方说法：

模块是一个包含Python定义和语句的文件。**文件名就是模块名后跟文件后缀 .py** 。在一个模块内部，模块名（作为一个字符串）可以通过全局变量__name__的值获得。

https://docs.python.org/zh-cn/3/tutorial/modules.html?highlight=__name__#modules

```
# !/usr/bin/python3
# -*-coding:UTF-8-*-
# FileName: Hello.py

def f(txt=None):
    if txt == None:
        txt = ''
    print(f'Hello {txt}!')
    print(f"__name__ == '{__name__}'")

if __name__ == '__main__':
    f()
```

Read More

全部讨论 回复

"if __name__=='__main__':" 的解释
ywjco_567 created at 3 days ago, Last updated at 3 days ago

1、如果当前模块是主模块（也就是调用其他模块的模块），那么此**模块名字就是__main__**，通过if判断这样就可以执行"__mian__:"后面的主函数内容；

2、如果此模块是被import来的，则此**模块名字为文件名字（不加后面的.py）**，通过if判断这样就会跳过"__mian__:"后面的内容。

```
# 伪造的代码:

if run(self, self.model): __name__=='__main__'
if run(import, import.model): __name__== import.model.__name__
```

全部讨论 回复

模块里非函数变量的引用
阿你烦不烦 created at May 9, 2019 8:03 PM, Last updated at 3 days ago

模块：

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

' a test module '

__author__ = 'Michael Liao'

i=5
```

主程序：

```
import testi=7print(test.i)print(i)
```

输出：

5

7

想请问下这种要如何理解，模块和主程序中的i指向不一样吗

> 阿你烦不烦
> Created at May 9, 2019 8:05 PM
>
> 主程序是(不知道为什么变成一行了)：
>
> import test
>
> i=7
>
> print(test.i)
>
> print(i)

> 郭大侠14
> Created at May 19, 2019 12:31 PM
>
> print(test.i) 输出的是test模块中的i啊，test模块中 i = 5； print(i) 输出的是程序里定义的i, i=7

> ywjco_567
> Created at 3 days ago
>
> test.i是明确指定使用test模块里面定义的i变量；
>
> i---是用本函数里定义的变量；
>
> 所以，test.i 与 i 是两个不同的变量。

全部讨论 回复

java调用python？
Meteor_0723 created at May 22, 2019 2:49 PM, Last updated at May 22, 2019 2:49 PM

想问下大家都是通过什么方式实现：java调用python的？

全部讨论 回复

使用模块
鸽王草咕咕 created at February 11, 2019 12:01 PM, Last updated at May 12, 2019 1:07 PM

写模块的格式： #!/usr/bin/env python3 # -- coding: utf-8 -- （标准格式）

' a test module '#对模块的注释

__author__ = 'Michael Liao'#模块作者的姓名

def test(): ... if __name__ == '__main__' test()#检查模块姓名，如果运行的模块叫这个名字，就输出这个模块的结果，不然只输出当前模块的运用结果。如果被引用的模块中没有这一行，那么运行时，会显示被引用模块的运行结果，以及当前模块的运行结果。

我们简单的理解就是： 如果模块是被直接运行的，则代码块被运行，如果模块是被导入的，则代码块不被运行。

作用域：有下划线的函数或变量名时非公开的（private），类似_xxx和__xxx，仅仅在模块内部使用，无法被直接引用。

> Wenshuang_PlsTryAgnLtr
> Created at February 18, 2019 5:27 PM
>
> 有下划线的函数或变量名时非公开的（private），类似_xxx和__xxx，仅仅在模块内部使用，无法被直接引用。
>
> 这里准确说应该是不建议直接引用？实际上直接调用私有函数也是可以成功的，比如：
>
> import hello
>
> hello._private_1('James')

> 拿笔小星
> Created at May 12, 2019 1:07 PM
>
> python 使用不同方法导入模块，模块中私有变量的使用区别

全部讨论 回复

在主文件定义一个全局变量，不传参的情况下如何让其他模块可以访问？
O3O——OTO created at March 1, 2019 3:04 PM, Last updated at May 1, 2019 10:33 PM

在主文件定义一个全局变量，不传参的情况下如何让其他模块可以访问？

比如main.py中定义一个变量new_var = 5

想在model.py文件中直接使用

> 廖雪峰
> Created at March 6, 2019 2:00 PM
>
> ```
> import hello
> hello.test() # test是hello模块的一个函数
> ```
>
> 变量和函数并没有本质区别，函数名就是变量
>
> ```
> import hello
> hello.new_var # new_var是hello模块的一个变量
> ```

> Hk_Mayfly
> Created at March 6, 2019 2:04 PM
>
> 我觉得你可以定义一个全局变量模块，在main.py和model.py中导入使用

> 喜欢游水的旱鸭子
> Created at May 1, 2019 10:33 PM
>
> 了解

全部讨论 回复

"if __name__ == '__main__': " 实现的功能就是Make a script both importable and executable
又得换昵称了 created at April 26, 2017 8:55 PM, Last updated at May 1, 2019 10:29 PM

关于代码if **name** == '**main**': 可能看了之后可能挺晕的，下面举几个例子解释下，希望能让大家对这行的代码的功能有更深的认识，还是那句话，欢迎大家指正定会虚心接受。

先编写一个测试模块atestmodule.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

' a test module '

def addFunc(a,b):
    return a+b

print('atestmodule计算结果:',addFunc(1,1))
```

再编写一个模块anothertestmodule.py来调用上面的模块：

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

' a test module '
```

Read More

> 又得换昵称了
> Created at April 26, 2017 9:01 PM
>
> 不知道怎么的，变量两边的下划线就是显示不出来，可能和格式冲突？
>
> 只有在代码中注释才会正常显示两边的下划线，只要不放在插入的代码中无论是在标题中还是正文中呈现出来的都没有了两边的下划线，所以我就把正文中的部分文字放到了代码中以注释的方式呈现出来，标题我就无能为力了，这里解释一下。
>
> 另外有谁知道这要怎么解决么？

> 又得换昵称了
> Created at April 26, 2017 9:02 PM
>
> 在讨论界面又可以呈现出来，也是蛮不懂得，希望不要影响到大家

> 用户5750871805
> Created at May 2, 2017 6:15 PM
>
> 弄懂这个只需要在google上搜索： python if **name** == '**main**
>
> 然后看stack overflow上的第一个回答就知道了。

> 千叶哲
> Created at May 3, 2017 1:59 PM
>
> 太谢谢楼主了，跟着你的步伐前进中，万语千言抵不过一句：谢谢！

> 跳怪003
> Created at May 5, 2017 2:50 PM
>
> 感谢楼主的解释，不然还真没信心看下去了

> AC使我快乐_
> Created at May 8, 2017 7:56 AM
>
> Thanks

> PeaceHeart_Liang
> Created at May 8, 2017 1:54 PM
>
> 想的太复杂了。。。下面是我的理解。学过C语言的都知道，C语言定义主程序入口不就是main()函数吗，main代表了程序主入口，即和系统的接口（说白了就是命令行直接调用）。if **name** == '**main**': 这句话的核心无非就是在判断该程序文件是否作为主程序入口罢了。如果在命令行直接调用该程序文件，该文件作为主程序入口，**name** == '**main**'理所当然啊。如果在命令行调用其他程序文件，主程序入口**name** == '**main**'自然不成立，因为**main**等于那个你在命令行输入的程序名。这个东西的好处就是，别人调用时（你并非是主程序入口）后面的东西不运行，自己命令行执行时（你是主程序入口）后面的东西运行。故可以作为测试用。

> 栉风32092
> Created at May 11, 2017 9:40 PM
>
> 谢，楼上各位大佬

> 鹿与陆
> Created at May 22, 2017 2:36 PM
>
> 楼主好厉害

> 单纯_喜欢
> Created at May 25, 2017 7:46 PM
>
> 谢谢以上各位大佬

全部讨论 回复

J
清纯超人 created at April 10, 2019 9:20 PM, Last updated at April 10, 2019 9:20 PM

xx

全部讨论 回复

if __name__ =="__main__": 的使用，还是挺详细的，希望各位学友能看懂
一身诗意喂了狗Sh created at March 29, 2019 9:30 AM, Last updated at March 29, 2019 9:30 AM

其实if **name** =="**main**"就相当于java里的 public static void main(String[] args) {

```
    方法体

}
```

都是函数执行的入口，比如： 1、先定义了一个函数sum(): def sum(a,b):
print("%d+%d=%d"%(a,b,a+b)) 2、再在下面进行调用
if **name** =="**main**": sum(1,3) 或者再改造一下：以后项目中可能调用的方法会很多，都写在上面会很乱，可以这样：

新定义一个main()函数，再在if **name** =="**main**"中去调用，如下： def sum(a,b): #两数之和的函数 print("%d + %d=%d"%(a,b,a+b)) def num(a,b): #两数乘积的函数 print("%d %d=%d"%(a,b,ab)) def main():
#在main()函数中分别调用上面2个方法 sum(2,3)
num(4,5) if **name** =="**main**": main() #调用main()函数

全部讨论 回复

python中__name__的使用
青铜神奢 created at March 16, 2019 8:44 PM, Last updated at March 18, 2019 8:20 PM

转僵尸认识你转载自https://www.cnblogs.com/1204guo/p/7966461.html

这几天开始学习Python，遇到一些问题，也解决了一些问题。

其中if **name** == '**main**'：这句估计很多和我一样的初学者都是不求甚解。这里作一下解释：

1：name是一个变量。前后加了爽下划线是因为是因为这是系统定义的名字。普通变量不要使用此方式命名变量。 2：Python有很多模块，而这些模块是可以独立运行的！这点不像C++和C的头文件。 3：import的时候是要执行所import的模块的。 4：name就是标识模块的名字的一个系统变量。这里分两种情况：假如当前模块是主模块（也就是调用其他模块的模块），那么此模块名字就是main，通过if判断这样就可以执行"mian:"后面的主函数内容；假如此模块是被import的，则此模块名字为文件名字（不加后面的.py），通过if判断这样就会跳过"mian:"后面的内容。

通过上面方式，python就可以分清楚哪些是主函数，进入主函数执行；并且可以调用其他模块的各个函数等等。

> 刘烨超351929724
> Created at March 18, 2019 8:20 PM
>
> 为什么我运行hello.test()结果是Too many arguments!

全部讨论 回复

python中__name__的使用
僵尸认识你 created at March 9, 2019 8:45 AM, Last updated at March 9, 2019 8:45 AM

转载自https://www.cnblogs.com/1204guo/p/7966461.html

这几天开始学习Python，遇到一些问题，也解决了一些问题。

其中if __name__ == '__main__'：这句估计很多和我一样的初学者都是不求甚解。这里作一下解释：

1：**name**是一个变量。前后加了爽下划线是因为是因为这是系统定义的名字。普通变量不要使用此方式命名变量。 2：Python有很多模块，而这些模块是可以独立运行的！这点不像C++和C的头文件。 3：import的时候是要执行所import的模块的。 4：**name**就是标识模块的名字的一个系统变量。这里分两种情况：假如当前模块是主模块（也就是调用其他模块的模块），那么此模块名字就是**main**，通过if判断这样就可以执行"**mian**:"后面的主函数内容；假如此模块是被import的，则此模块名字为文件名字（不加后面的.py），通过if判断这样就会跳过"**mian**:"后面的内容。

通过上面方式，python就可以分清楚哪些是主函数，进入主函数执行；并且可以调用其他模块的各个函数等等。

全部讨论 回复

运行python hello.py 报错
asade69552 created at December 18, 2018 3:22 PM, Last updated at March 7, 2019 10:46 AM

我把例子里的代码复制到python解释器里，运行python hello.py为什么会报错，提示invalid syntax？

> Wafcc9
> Created at December 21, 2018 5:37 PM
>
> 复制粘贴的有时候不行，空格和制表粘贴的时候可能会乱

> bxy晨风雨
> Created at March 7, 2019 10:46 AM
>
> 问一下你解决这个问题了吗，我也有这个问题 win系统了例子里的代码复制到python解释器里，运行python hello.py为什么会报错，提示invalid syntax？

全部讨论 回复

为什么要加$
神气的圆圆 created at August 22, 2018 3:00 PM, Last updated at February 27, 2019 2:59 PM

python hello.py可运行前面加$会报错

> wxweven520
> Created at August 22, 2018 10:28 PM
>
> 建议你先去补下Linux的基本知识，『$』并不是说你自己敲上去的，你用普通用户登录后，Linux命令行最前面会自动显示『$』符号，就像你用root用户登录后，命令行最前面会显示『#』符号。所以，忽略这个『$』就可以了！

> 君陌归来
> Created at November 28, 2018 5:56 PM
>
> 哈哈哈，看到这里我笑出了声

> 月半文一刀
> Created at December 17, 2018 2:55 PM
>
> hahaha

> 月半文一刀
> Created at December 17, 2018 3:09 PM
>
> 可以理解，我刚开始学Linux的时候，也挺纳闷的~

> asade69552
> Created at December 18, 2018 7:24 PM
>
> 为什么我运行python hello.py 运行会报错

> DeepEndBoo
> Created at January 31, 2019 4:38 PM
>
> 哈哈哈$是linux下，终端上，普通用户的命令提示符。哈哈哈

> 时间里的D
> Created at February 20, 2019 2:43 PM
>
> 因为这是linux系统下的，不是windows系统下的

> 喜欢rap的hacker
> Created at February 26, 2019 1:23 PM
>
> 哈哈哈哈哈，弟弟啊，那是root权限下的linux的终端。

> tin猪麻麻
> Created at February 27, 2019 1:55 PM
>
> 哈哈，你前面加个# 就能运行了
>
> 开玩笑的那个是类linux 下的权限标识符$是USER #是ROOTUSER

> epad
> Created at February 27, 2019 2:59 PM
>
> 系统的shell 脚本

全部讨论 回复

另外，Python交互环境下import hello，廖大应该省略了很多步骤吧
Gin阿金 created at February 9, 2019 10:43 PM, Last updated at February 9, 2019 10:43 PM

具体是否可以参考https://blog.csdn.net/alanconstantinelau/article/details/68952256 ？(贴个站外链接应该可以吧？（小声比比）)

全部讨论 回复

运行python3 hello.py Michael获得的sys.argv就是['hello.py', 'Michael]
Gin阿金 created at February 9, 2019 9:52 PM, Last updated at February 9, 2019 10:39 PM

这句如何理解?.py Michal要怎么执行呢？win环境下另外sys.argv变量作用还是没有很明白，希望大佬解答一下

> Gin阿金
> Created at February 9, 2019 10:39 PM
>
> 自己尝试了一下，理解了：sys.argv读取的是命令行的参数，且必定包含文件路径，后面的Micheal是CMD下执行.py 后缀加上Micheal，即CMD下输入hello.py Micheal

全部讨论 回复

if __name__=='__main__': test()
Gin阿金 created at February 9, 2019 9:51 PM, Last updated at February 9, 2019 9:51 PM

if __name__=='__main__': test()

个人理解：只是用来区分是直接运行还是导入，直接运行需要显示结果，导入时不需要：直接运行时__name__值是__main__,所以直接运行test()显示结果导入时，一般时需要用的时候再调用，不需要显示，且此时__name__值非__main__，所以不会执行test() 具体main的含义参照楼上大佬的解释比较好理解

全部讨论 回复

有一个问题
333liuyi created at December 13, 2018 5:50 PM, Last updated at January 31, 2019 4:56 PM

请问一下，在导入hello模块后，调用hello.test()时，还能像"python hello.py Michel"一样方便添加参数Michel吗？

> DeepEndBoo
> Created at January 31, 2019 4:56 PM
>
> 不能。在交互模式中sys.argv无法获取到list了

全部讨论 回复

打卡
他国行尸 created at December 31, 2018 7:16 PM, Last updated at December 31, 2018 7:16 PM

没有protected权限

全部讨论 回复

会java的这一章过目一下就行了
教育不良 created at December 17, 2018 5:31 PM, Last updated at December 17, 2018 5:31 PM

嘿嘿

全部讨论 回复

发表评论

登录后发表评论