

INDEX



- Python教程
- Python简介
- 安装Python
- 第一个Python程序
- Python基础
- 函数
- 高级特性
- 函数式编程
- 模块
- 使用模块
- 安装第三方模块
- 面向对象编程
- 面向对象高级编程
- 错误、调试和测试
- IO编程
- 进程和线程
- 正则表达式
- 常用内建模块
- 常用第三方模块
- virtualenv
- 图形界面
- 网络编程
- 电子邮件
- 访问数据库
- Web开发
- 异步IO
- 实战
- FAQ
- 期末总结

模块

Reads: 27223020

在计算机程序的开发过程中，随着程序代码越写越多，在一个文件里代码就会越来越长，越来越不容易维护。

为了编写可维护的代码，我们把很多函数分组，分别放到不同的文件里，这样，每个文件包含的代码就相对较少，很多编程语言都采用这种组织代码的方式。在Python中，一个.py文件就称之为一个模块（Module）。

使用模块有什么好处？

最大的好处是大大提高了代码的可维护性。其次，编写代码不必从零开始。当一个模块编写完毕，就可以被其他地方引用。我们在编写程序的时候，也经常引用其他模块，包括Python内置的模块和来自第三方的模块。

使用模块还可以避免函数名和变量名冲突。相同名字的函数和变量完全可以分别存在不同的模块中，因此，我们自己在编写模块时，不必考虑名字会与其他模块冲突。但是也要注意，尽量不要与内置函数名字冲突。点[这里](#)查看Python的所有内置函数。

你也许还想到，如果不同的人编写的模块名相同怎么办？为了避免模块名冲突，Python又引入了按目录来组织模块的方法，称为包（Package）。

举个例子，一个`abc.py`的文件就是一个名字叫`abc`的模块，一个`xyz.py`的文件就是一个名字叫`xyz`的模块。

现在，假设我们的`abc`和`xyz`这两个模块名字与其他模块冲突了，于是我们可以通过包来组织模块，避免冲突。方法是选择一个顶层包名，比如`mycompany`，按照如下目录存放：

```
mycompany
├── __init__.py
├── abc.py
└── xyz.py
```

引入了包以后，只要顶层的包名不与别人冲突，那所有模块都不会与别人冲突。现在，`abc.py`模块的名字就变成了`mycompany.abc`，类似的，`xyz.py`的模块名变成了`mycompany.xyz`。

请注意，每一个包目录下面都会有一个`__init__.py`的文件，这个文件是必须存在的，否则，Python就把这个目录当成普通目录，而不是一个包。`__init__.py`可以是空文件，也可以有Python代码，因为`__init__.py`本身就是一个模块，而它的模块名就是`mycompany`。

类似的，可以有多个目录，组成多级层次的包结构。比如如下的目录结构：

```
mycompany
├── web
│   ├── __init__.py
│   ├── utils.py
│   └── www.py
├── __init__.py
├── abc.py
└── utils.py
```

文件`www.py`的模块名就是`mycompany.web.www`，两个文件`utils.py`的模块名分别是`mycompany.utils`和`mycompany.web.utils`。

⚠ 自己创建模块时要注意命名，不能和Python自带的模块名称冲突。例如，系统自带了sys模块，自己的模块就不可命名为sys.py，否则将无法导入系统自带的sys模块。

`mycompany.web`也是一个模块，请指出该模块对应的.py文件。

总结

模块是一组Python代码的集合，可以使用其他模块，也可以被其他模块使用。

创建自己的模块时，要注意：

- 模块名要遵循Python变量命名规范，不要使用中文、特殊字符；
- 模块名不要和系统模块名冲突，最好先查看系统是否已存在该模块，检查方法是在Python交互环境执行`import abc`，若成功则说明系统存在此模块。

读后有收获可以支付宝请作者喝咖啡，读后有疑问请加微信群讨论：



还可以分享给朋友：

分享到微博

Comments

Make a comment

Sign in to make a comment

