2.4 函数与模块

2023 年 9 月 8 日

函数与模块

一、函数

在 Python 中，定义一个函数要使用 `def` 语句，依次写出函数名、括号、括号中的参数和冒号:，然后，在缩进块中编写函数体，函数的返回值用 `return` 语句返回。

```
def 函数名 (参数1, 参数2, ···):    # 注意冒号是语法必备元素
    函数语句块                      # 缩进表明语句块内的所有语句都在本函数作用域下
    [return 表达式].               # 方括号表示可能有返回值，也可能没有返回值
```

我们以自定义一个求绝对值的 `my_abs` 函数为例:

[1]:
```
def my_abs(x):
    if x >= 0:
        return x
    else:
        return -x
```

现在我们调用它看看:

[3]:
```
my_abs(-99)
```

[3]:
```
99
```

练习

请设计一个比较大小的函数，compare(x, y)，使用 return 语句返回两者中较大的值。

[ ]:
```

```

二、模块

Python 最高级别的代码组织单元是模块（Module）。Python 可以将实现某种特定功能的代码分离出来，定义成函数，并放在模块文件里。

每个以.py 为后缀名的 Python 源代码文件就是一个模块，Python 模块可以通过 import 语句导入另一个模块中。

模块分为标准库模块，第三方模块和自定义模块。

标准库是 python 安装文件内置的满足日常编程使用的库，它包含以下模块，见下表。这里只列举了部分，更多模块请见官方文档: 链接

模块	功能
os	提供了许多与操作系统交互的函数
datetime	模块提供了以简单和复杂的方式操作日期和时间的类
re	为高级字符串处理提供正则表达式工具
random	模块提供了进行随机选择的工具
doctest	用于扫描模块并验证程序文档字符串中嵌入的测试

第三方模块是有第三方组织和个人提供的实现特定功能的模块。| 模块 | 功能 | | ———————————————————— | ———————————————————— | | numpy | 提供了数组、矩阵和数值计算功能 | | pandas | 提供了强大、便捷的数据分析工具 | | scipy | 提供了科学计算、统计等相关工具 | | statistics | 模块计算数值数据的基本统计属性（均值，中位数，方差等）| |statsmodels | 提供了多元回归和分析的工具 | | sklearn | 提供了机器学习、数据挖掘模型 |

2.1 import 模块名

在程序里，当使用模块时，我们以关键字 `import` 开头，加上模块的名字。我们以 datetime 模块为例，datetime 模块，提供当日的日期，试运行以下代码：

[4]:
```
import datetime
```

[5]:
```
datetime.datetime.today()
```

[5]:
```
datetime.datetime(2023, 3, 5, 21, 42, 36, 159372)
```

2.2 from 模块名 import 函数名

有些情况下，程序并不需导入整个模块，而是仅仅导入程序要用到的函数，此时可以使用以下方式导入具体的函数。

[6]:
```
from math import sqrt
sqrt(2)
```

[6]:
```
1.4142135623730951
```

练习

1. 在当前目录下，点击右上方蓝色 □ 号，新建一个 Python 文件
2. 在文件中放入上一练习的函数 compare(x, y)，保存文件名称为 my_func.py
3. 在下方空格处，导入该文件中的 compare 函数
4. 调用 compare 函数，比较 2 的 33 次方和 3 的 22 次方的大小。

[ ]:
```

```

3.3 安装第三方库

我们使用 **pip** 这个工具安装第三方库，在 shell 命令行环境下，输入：

```
% pip install 库名
```

显示库信息：

```
% pip show 库名
```

更新库：

```
% pip install --upgrade 库名
```

更新库到指定版本：

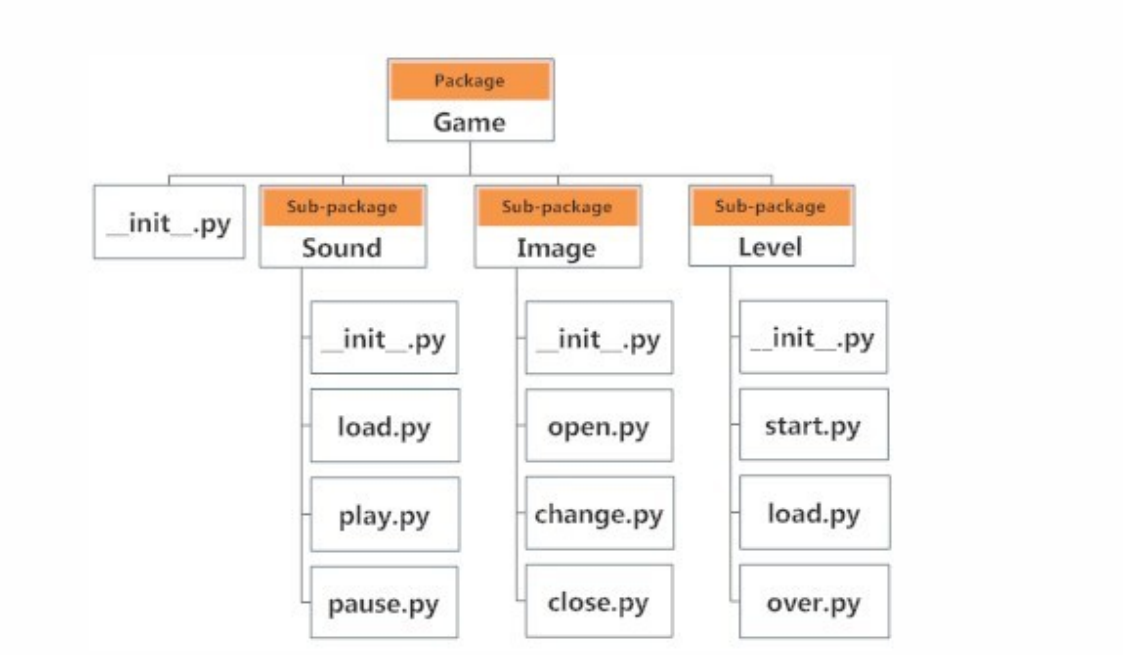
```
% pip install --upgrade 库名==版本号
```

我们访问 pypi 网站链接，在搜索栏中输入你想要的关键字，找到心仪的，然后使用 **pip** 工具在你的电脑 shell 命令行下安装就可以了。

三、包的概念

什么是**包** (Package) 呢？简单来说，**包**是包含模块的文件夹。**模块**是我们的以.py 为后缀的 python 文件。

以下图为例子，这是一个和游戏相关的程序，它的文件结构如下：



这里 Game 文件夹叫包（Package），Sound、Image 和 Level 这三个文件夹叫子包 (Sub-package)。它们各自包含自己的.py 文件，这些文件叫模块。

从这里我们可以看到，包是一个有层次的文件目录结构，它定义了由 n 个模块或 n 个子包组成的 python 应用程序执行环境。通俗一点：包是一个包含 __init__.py 文件的目录，该目录下一定得有这个 __init__.py 文件和其它模块或子包。

注意，每一个包中的 `__init__.py` 文件是必须存在的，否则，Python 就会把这个目录当成普通目录(文件夹)，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，而它的模块名就是对应包的名字。