Python 基础

金　林
中南财经政法大学统计系
jinlin82@qq.com

2020 年 1 月 7 日

Python Logo

Python 之父

Guido van Rossum, the creator of Python

基本情况

1. 设计者：Guido van Rossum,
2. 发行时间：1991.2.20; 29 年前
3. 稳定版本：
   1. 3.8.1 /2019 年 12 月 18 日
4. 操作系统：跨平台
5. 许可证：免费和开源软件
6. 官网: www.python.org
7. 之所以选择 python 作为程序的名字，是因为设计者 Guido 心情不佳 (同时还因为 Guido 是英国肥皂剧《Monty Python 飞行马戏团》的粉丝)

特点与理念

1. Python 是一种 多范式 编程语言:
   1. 面向对象的程序设计
   2. 结构化程序设计
   3. 而且许多特性都支持函数式编程
2. 该语言的核心理念在文档 The Zen of Python（PEP 20）中进行了概述，其中包括如下格言：
   1. 美丽胜于丑陋
   2. 明了胜于晦涩
   3. 简单胜于复杂
   4. 复杂胜于凌乱
   5. 可读性很重要

安装

1. 安装使用 Anaconda
2. Anaconda 是可以便捷获取包且对包能够进行管理，同时对环境可以统一管理的发行版本。Anaconda 包含了 conda、Python 在内的超过 180 个科学包及其依赖项。
3. 网站：`https://www.anaconda.com/`
4. 国内镜像：`https://mirror.tuna.tsinghua.edu.cn/help/anaconda/`

- 控制台
- 集成开发环境

Python.exe–交互模式

1. 当命令是从 tty 中读取时，称解释器处于交互模式。
2. 在这个模式中它根据主提示符执行命令, 通常是三个大于号 (`>>>`);
3. 为延续行提供了从属提示符，缺省是三个点 (...).
4. 输入多行结构时需要延续行

IPython

- IPython 提供了一个丰富的工具包，可帮助您充分利用 Python 进行交互。它的主要组成部分是:
  1. 强大的 python 交互式 shell
  2. Jupyter 内核，可在 Jupyter 笔记本和其他交互式前端中使用 Python 代码

- 控制台
- 集成开发环境

集成开发环境

1. IDLE
2. Spyder
3. Vscode
4. Emacs
5. PyCharm
6. 其它

模块 (Module)

1. 模块是一个包含一些 Python 定义和语句的文件
2. 文件名就是模块名后面添加后缀.py

包 (Package)

1. 包是一种通过使用“. 模块名”来构造 Python 模块命名空间的方式。
2. 例如，模块名 A.B 表示 A 包中名称为 B 的子模块。

库 (Library)

- Python 有一个大型的标准库，被认为是 Python 的最大优势之一，提供了适合许多任务的工具
- 复杂且功能强大，被描述为“功能齐全”的 Python 哲学
- 对于面向互联网的应用程序，支持许多标准格式和协议（例如 MIME 和 HTTP）。
- 包含用于创建图形用户界面，连接到关系数据库，伪随机数生成器，精确到任意小数的算术，操作正则表达式以及进行单元测试的模块。

Python 中用于科学计算和人工智能常见库

1. 基本库
   1. math
   2. Numpy
   3. SciPy
   4. Pandas
2. 统计模型
   1. statistics
   2. statsmodels
3. 图形和可视化
   1. matplotlib
   2. seaborn
   3. ggplot2

Python 中用于科学计算和人工智能常见库

4. 机器学习
   1. scikit-learn
   2. xgboost
5. 深度学习和人工智能
   1. TensorFlow
   2. Keras: The Python Deep Learning library
   3. Pytorch
6. 大数据
   1. PySpark

模块导入

1. `import module1, module2`
   - 通过模块名加点 (.) 访问函数

```
import math
math.sin(3)
"""""""
```

2. `import modname as mm`
3. `from modname import name1, name2`
4. `from modname import *`

PyPI - Python 包索引

- 截至 2019 年 12 月，Python 软件包索引 (包含用于 Python 的第三方软件的官方存储库) 包含 211,000 多个项目，可提供广泛的功能，其中包括：
  1. 图形用户界面、网络框架、多媒体、数据库、网络和通信
  2. 测试框架、自动化和 web 抓取、文档工具、系统管理
  3. 科学计算，文本处理，图像处理

dir() 函数

1. 内置函数 dir() 用于查找模块定义的名称。
2. 不带参数的 dir () 列出当前定义的名称:

查看 modules

- 查看内置函数 (builtin 模块中的函数)：`dir(__builtin__)`
- 查看内置模块 (written in C and built in to the Python interpreter)：`sys.builtin_module_names`
- 查看标准模块：
  1. CMD 中 `pip install stdlib-list`
  2. `>>> from stdlib_list import stdlib_list`
  3. `>>> libraries = stdlib_list("2.7")`
- 查看所有安装的模块：`help("modules")`，CMD 中 `pip list`
- 查看当前加载的模块：`dir()`; `sys.modules.keys()`

查看某一 package 中所有子包和子模块

- 使用标准库中的 `pkgutil` 库，如列出 `numpy` 中的所有子包 (sub packges) 和子模块 (sub modules)

```
1 import pkgutil
2 import numpy
3
4 for importer, modname, ispkg in pkgutil.iter_modules(numpy.__
      path__, prefix="numpy."):
5     print(modname)
6
7 """""""
```

简介

1. 使用函数组织代码，使用内置类型组织数据
2. “面向对象编程”：使用程序员定义的类型来组织代码和数据。
3. Python 内置类型:
   1. list
   2. tuple
   3. dict
   4. set

自定义的类型：class

1. 自定义的类型也称为 class. 类的定义类似如下:

```
class Point:
    """Represents a point in 2-D space."""
Point()
"""""""
```

2. 开头表示创建一个新类 Point
3. 主体是一个字符串，用于解释该类的作用
4. 可以在类定义中定义变量和方法

对象

1. 定义一个名为 Point 的类会创建一个类对象:

```
>>> Point
<class '__main__.Point'>
```

2. 类对象就像一个创建对象的工厂。
3. 要创建 Point，可以将 Point 当作函数来调用

```
>>> blank = Point()
>>> blank
<__main__.Point object at 0xb7e9d3ac>
```

实例化

1. 返回值是对 Point 对象的引用，我们将其指定为空。
2. 创建一个新对象称为 实例化, 这个对象是类的 实例
3. 当你输出一个实例时，Python 会告诉你它属于哪个类以及存储的位置
4. (前缀 0x 表示数字为十六进制)
5. 每个对象都是某个类的实例，因此“对象”和“实例”是可以互换的。

属性

1. 实例后加点 (.) 对变量进行赋值

```
>>> blank.x = 3.0
>>> blank.y = 4.0
```

2. 这些元素称为属性。
3. 使用相同的语法读取属性值：

```
>>> blank.y
4.0
>>> x = blank.x
>>> x
3.0
```

函数

1. 纯函数
   - 纯函数的返回结果只依赖于它自己的参数，函数执行过程里面没有副作用，输入相同的参数，

   返回同样的结果
2. 修饰符
   - 修改作为参数获取的对象的函数。

方法

- 方法 是与特定类关联的函数。

```
a=[1,2,3]
import numpy as np
a=np.array(a)
a+1
"""""""
```

- 方法在绑定后立即被调用：

```
x.f()
```

方法和函数

- 方法在语义上与函数相同，但是在语法上有两个区别：
  1. 在类定义内定义方法，以使类与方法之间的关系明确。
  2. 调用方法的语法与调用函数的语法不同。
- 方法和函数的形式可以相互更改，可以根据自己的工作选择最佳形式。

魔术方法

1. 它们是具有固定名称的特殊方法。它们具有特殊的语法，以"_"(双下划线) 作为名字的开头和结尾
2. 不用直接调用魔术方法。调用是在后台实现的。
3. 按照惯例，方法的第一个参数称为 self ，第二个参数称为 other 。
4. init 方法（“initialization” 的缩写）是一种特殊的方法，在实例化对象时会调用该方法。
5. str 是一种特殊的方法，如 init 一样，应该返回对象的字符串表示形式（带有 `print` 函数）。

运算

1. 可以使用 `**` 运算符来计算幂运算
2. 在交互模式下，最后输出的表达式分配给变量 _ 。

缩进

1. 缩进是 Python 对语句进行分组的方式。
2. 在交互式提示下，您必须为每个缩进的行键入一个制表符或空格。
3. 在实践中，使用文本编辑器为 Python 准备更复杂的输入。所有较好的文本编辑器都具有自动缩进功能。
4. 以交互方式输入复合语句时，必须在其后跟随一个空行以表示完成（因为解析器无法猜测何时键入了最后一行）。
5. 请注意，基本块中的每一行都必须缩进相同的大小。

代码风格：PEP8

1. 大多数语言可以用不同的风格来书写（或者更简洁，更格式化）；有些语言比其他更具可读性。.
2. 对于 Python，PEP8 已成为大多数项目所遵循的样式指南。它是一种非常易读且令人赏心悦目的编码风格。.