最近在很多地方都可以看到Python的身影，尤其在人工智能等科学领域，其丰富的科学计算等方面类库无比强大。很多身边的哥们也提到Python非常的简洁方便，比如用**Django**搭建一个见得网站只需要半天时间即可，因此也吸引了我不小的兴趣。之前相亲认识过一个姑娘是做绿色环保建筑设计行业的，提过她们的建筑物的建模也是使用Python，虽然被女神给拒绝了，但学习还是势在必行的，加油。

这部分只涉及python比较基础的知识，如复杂的面向对象、多线程、通信等知识会放在之后的深入学习中介绍，因此整个学习过程也将非常的快速，3-5个小时完全足够了。



Python是一种[面向对象](http://baike.baidu.com/view/125370.htm)、解释型[计算机程序设计语言](http://baike.baidu.com/view/2561555.htm)，由[Guido van Rossum](http://baike.baidu.com/view/2975166.htm)于1989年发明，第一个公开发行版发行于1991年。是纯粹的[自由软件](http://baike.baidu.com/view/20965.htm)，[源代码](http://baike.baidu.com/subview/60376/5122159.htm)和[解释器](http://baike.baidu.com/view/592974.htm)CPython遵循[GPL](http://baike.baidu.com/view/130692.htm)协议。其设计哲学是优雅、明确、简单，最NX的是其专用的科学计算扩展库，NumPy、SciPy和matplotlib等，可以很大程度的替代MATLAB的作用。作为一种解释性语言，其源码通过虚拟机编译为字节码文件，通常为.pyc格式。其通常版本包括2.x和3.x，前者仍然是主流，后者正在不断发展中，本部分的学习将使用2.x版本。此外，比较有意思的一点是，Python是强制缩进的语言，如果换行后不缩进，会报错。

Python版本选用2.7.11，下载地址：<https://www.python.org/downloads/>

集成开发环境使用：Sublime，下载地址：<http://www.sublimetext.com/>。

此外，在另一篇博文中：xxx，介绍了如何搭建在sublime中安装插件，构建一个足够好用的python工作环境。

* **变量**

|  |  |  |
| --- | --- | --- |
| 基本数据类型 | 存储内容 | 示例 |
| integer(int) | 整数 | 1,6,-3 |
| float | 浮点数 | 3.14 |
| long | 非常大的整数 | 100000004 |
| string | 字符串 | “Hello World” |
| list | 列表 | [1,2,3] |
| tuple | 元组，广义表 | {“hello”, 5} |
| dictionary | 字典 | {‘name’:’xionger’,’grade’:2} |

常见的运算符包括：+,-,\*,/,%;特殊的有：//,Floor除法;-,取反;abs()，绝对值;\*\*,求幂等。

常见的比较运算包括：==,!=,>,<,>=,<=。

**字符串常见操作**

|  |  |
| --- | --- |
| 函数 | 功能 |
| .upper()/lower() | 把所有字母转化为大写/小写 |
| .capitalize() | 把字符串首字母答谢，其他小写 |
| .title() | 把首字母和每个空格或标点符号后的字母大写，其他小写 |
| =, +, \* | 字符串的赋值，拼接，重复 |
| == | 字符串的比较 |
| \n, \\ | 转义字符 |
| .strip(),.rstrip(),.lstrip() | 去除字符串两端空格，或指定字母 |
| .() | 字符串长度 |
| .find() | 查找子串，返回第一个满足的位置索引 |
| .replace(“a”, “b”) | 字符串替代 |

Tip:一个关于utf-8的问题，添加之后注释即可：#! usr/bin/python #coding=utf-8

* **逻辑结构**

|  |  |
| --- | --- |
| 逻辑结构 | 示例 |
| 判断 | if a > 5:  print("great")  else:  print("equal or less") |
| 异常处理 | try:  5/0  except:  print("Please don't do that") |
| 循环 | for i in range(5):  if i == 0:  **continue**  if i == 3:  **break**  while not age.isdigit()  Print 'sorry'  While True |

* **集合对象**

**列表相关操作**

|  |  |
| --- | --- |
| 操作 | 解释或示例 |
| 创建列表 | fruit=['apple', 'strawberry', 'pear', 'papaya'] |
| .len(), .count() | 列表总项数；统计某一项的总数量 |
| .index() | 返回某项在列表中第一次出现的位置 |
| ‘pink’ in color\_list | 判断该项是否在列表中 |
| .append(),.extend() | 添加新项到列表；扩展列表，附加后一个列表到前一个列表 |
| Fruits[0] | 根据索引获得指定项 |
| .remove(xx),.insert(index, xx) | 根据索引删除指定项；插入某项到指定位置 |
| +, \* | 合并列表和复制列表成员 |
| .reverse(), .sort() | 反转列表；按字母排序列表 |
| ==, != | 列表比较 |
| range（7），range(-5,5,2) | 生成数字列表，参数分别为：初始值，结束值，步长 |

**字典相关操作**

|  |  |
| --- | --- |
| 操作 | 解释或示例 |
| 创建字典 | person = {'xionger':'2b', 'xiongda':'god'} |
| person['xionger'] | 根据key获取字典中的值 |
| .pop() | 弹出某项并从字典中删除 |
| .has\_key(), in | 判断该key是否存在 |
| .keys(), .values() | 通过列表，显示所有的key/value |

* **函数、类与对象（继承）**

一个简单的函数示例如下：

|  |
| --- |
| def get\_seat\_total(seat=1):  total=0  for dish in seat:  total = total + dish  return total |

一个简单的类和对象使用示例（**封装**）如下所示：

|  |
| --- |
| class Student(object):  def \_\_init\_\_(self, name, grade, distinct):  self.name =name  self.grade=grade  self.distinct=distinct  def printName(self):  print self.name  def main():  student01 = Student("xionger", 2, "SH")  student01.printName()  if \_\_name\_\_=="\_\_main\_\_":  main() |

类中常见可重写方法（**多态**）列表

|  |  |
| --- | --- |
| 可重写方法 | 解释 |
| \_\_init\_\_(self): | 构造函数，\_\_new\_\_也可以 |
| \_\_eq\_\_/\_\_ne\_\_(self, other): | 比较相等与不等 |
| \_\_gt[e]\_\_/\_\_lt[e]\_\_(): | 比较大小 |
| \_\_str\_\_: | Print输出，类似ToString() |

类**继承**的简单示例如下：

|  |
| --- |
| class Book(InventoryItem):  def \_\_init\_\_(self, title, price):  super(Book, self).\_\_init\_\_(title=title)  self.price=price |

Tip:由此可以看到，python完全符合面向对象的思想，实现了OO的封装、继承、多态三大特性。

* **输入输出与文件I/O**

|  |  |
| --- | --- |
| 函数 | 功能 |
| Input() | 收集输入信息 |
| Raw\_Input() | 收集任何非数字的信息 |
| Float(), int() | 转化为浮点数或整型 |
| Getpass() | 获取密码，需要导包 from getpass import getpass |
| print "Hello {word}".format(word="World") | 格式化输出 |

文件和目录相关IO操作（包括json）的示例如下所示：

|  |  |
| --- | --- |
| 操作 | 示例 |
| 从文件读取数据 | f=open('car.json')  car = f.readlines()  f.close() |
| 把数据写入文件 | f=open('car.json', 'r+w') #需要注意'w'会擦除所有数据，r+w会覆盖数据  f.write('test')  f.close() |
| 添加数据到文件 | f=open('car.json', 'a') #使用a表示append不会擦除数据，而是会添加到末尾  f.write('test') |
| 创建文件 | f=open('car.json', 'w+') |
| 文件列表 | current\_dir=os.getcwd(), os.listdir(current\_dir) |
| 在目录间移动 | class\_dirs=os.walk('.')  print class\_dirs.next() |
| 创建目录 | os.makedir(), os.makedirs() |
| 获取文件信息 | print os.stat('car.json') |
| 读取json文件 | import json  f=open('car.json')  car = json.load(f)  print type(car) |
| 打印json | print json.dumps(car, indent=2) |
| 把对象保存为json | json.dump(vars(stu), f, indent=2)，注意dump与dumps的区别 |
| 嵌套的对象转化为json | def get\_JSON\_dict(self):  d=vars(self)  student\_list=[]  for student in self.students:  student\_list.append(vars(student))  d['students']=student\_list  return d |

* **模块和文档**

模块导入方式： import module; from module import class; from module import function;from module import \*。

常见模块包括：random，生成随机数；os，用于与操作系统交互；json，用于生成和读取JavaScript Object Notation(JSON)；sqlite3，用于创建、编辑和读取sqlite3数据库；datetime，用于操作时间和日期；getpass，用于获取用户敏感信息的包。之后通过一个列表了解最常见的类库方法。

|  |  |
| --- | --- |
| 函数 | 解释或示例 |
| randint(a,b)/random(x)  /uniform(a, b) | 随机整数、随机浮点数（不包含0和1）  特定区间浮点数 |
| Choice(list) | 从列表中获取随机对象 |
| time() | Lunch=time(11,30),lunch.hour/mintue/second |
| datetime() | datetime.datetime(year=2016,day=14,month=4)  datetime.timedelta(days=14)  datetime.now()，当前时间 |

**文档结构：**其中\_\_init\_\_.py是空的，用于告诉python，需要使用这部分代码。

|  |
| --- |
| myproject/  -main.py  -classes/  -\_\_init\_\_.py  -monsters.py  readme.txt  install.txt |

比如，需要main.py中使用from classes.monsters import Monster。

docstring:通常用三引号、单引号括起来，可以在shell中通过help(xxx)来查看文档。

* **数据库**（sqlite）

Sqlite是一种超轻型的关系型数据库，支持ACID操作，常用语移动设备和浏览器中。其安装非常简单，地址：<https://www.sqlite.org/download.html>，下载windows下的[sqlite-tools即可，之后可以通过firefox的sqliteManager插件直接管理。](https://www.sqlite.org/2016/sqlite-tools-win32-x86-3130000.zip)

其数据类型很简单，包括：Null；nteger，整型；Real，浮点数；Text，字符串；Blob，一个文件，如图片、歌曲等。

|  |  |
| --- | --- |
| 操作 | 解释或示例 |
| 建表 | import sqlite3  conn=sqlite3.connect('mytest.sqlite')  cursor=conn.cursor()  sql='''create table students(  name text,  username text,  id int)'''  cursor.execute(sql)  cursor.close() |
| 插入操作 | sql='''insert into students(name, username, id)  values(:st\_name, :st\_username, :id\_num)'''  cursor.execute(sql, {'st\_name':'xionger', 'st\_username':'x2', 'id\_num':1322})  conn.commit()· |
| 查询操作 | sql="select \* from students"  results=cursor.execute(sql)  all\_students=results.fetchall() |

* **Debug**

通过TraceBack栈轨迹来查找错误，比如TypeError信息。还可以使用pdb调试器来查找错误，语句为:python –m pdb xxx.py，相关的pdb命令为：args,传输参数；break，增加断点；cont,继续知道下一个断点；clear，清除所有断点；list，显示当前所在位置附近的代码；next，执行代码的当前行；step，执行代码当前行，如果进入函数则停止。

这部分想说的是，在intelli IDE(pycharm)中调试非常方便，不过sublime中的调试还没弄清楚，之后补上。

* **Flask的Web开发**

这儿将介绍python最吸引人的部分，开源库的使用了，这部分主要介绍最为常见的Web框架Flask的相关知识，首先是安装步骤。

步骤1：安装setuptools,通过在https://pypi.python.org/pypi/setuptools下载ex\_setup.py文件，之后运行即可。

步骤2：安装pip,首先设置好之前下载的工具的环境变量，然后可以直接使用命令easy\_install pip。

步骤3：pip install flask，此外么，其他很多的python组件都可以通过该方式安装。

接下来就让我们通过一个简单的例子来看看flask构建一个动态网站有多简单吧。

|  |
| --- |
| from flask import Flask  from flask import render\_template  app=Flask(\_\_name\_\_)  @app.route('/')  def hellp\_world():  return 'Hello World!'  @app.route('/lucky/<number>/')  def lucky(number):  return render\_template('lucky.html', lucky\_num=number)  if \_\_name\_\_=='\_\_main\_\_':  app.run() |

Tip:里面用到**jinja**的视图模板，平时也可以用它来做**代码生成工具**，非常快速小巧，主要注意{{}}方式和angularJS一样，混用需要调整。

Python常见的组件介绍

|  |  |
| --- | --- |
| 组件名 | 解释或示例 |
| PyGame | 游戏和动画开发 |
| Django | Web框架，功能强大且灵活，比flask更复杂 |
| Jinja2 | 页面模板，也可以用作代码生成工具，很棒 |
| Plone | 功能完备的内容管理系统，直接可以使用 |
| Tkinter,Pyjs,PyGUI | 桌面应用库 |
| Kivy | iOS和Android应用 |
| NumPy,SymPy,SciPy Library,Matplotlib,pandas | 科学计算 |

Django的整个使用过程完全可以参考官方文档，<https://docs.djangoproject.com/en/1.9/intro/tutorial01/>，接下来跟着官方文档一起来创建一个最简单的应用。

|  |  |
| --- | --- |
| 步骤 | 代码 |
| 创建项目(站点) | django-admin startproject mysite  #项目的目录如下所示  manage.py：命令行工具  mysite/settings.py:配置文件  mysite/urls.py:url路由设置文件  mysql/wsgi.py:部署文件 |
| 查看项目（空） | python manage.py runserver |
| 创建指定应用 | python manage.py startapp polls |
| 写第一个页面 | **polls/views.py**  from django.http import HttpResponse  def index(request):  return HttpResponse("Hello, world. You're at the polls index.")  **polls/urls.py**  from django.conf.urls import url  from . import views  urlpatterns = [  url(r'^$', views.index, name='index'),  ]  **mysite/urls.py**  from django.conf.urls import **include**, url  from django.contrib import admin  urlpatterns = [  url(r'^polls/', include('polls.urls')),  url(r'^admin/', admin.site.urls),  ] #注意粗体的include不要忘记 |
| 查看项目polls应用 | 运行server，<http://localhost:8000/polls/> |
| 数据库设置(只介绍sqlite) | python manage.py migrate |

这部分内容就介绍到这儿为止，之后都是具体的模块建立了，详情请参考官方文档。

此外Jinja学习：官网<http://jinja.pocoo.org/>，中文站http://docs.jinkan.org/docs/jinja2/

预祝大家有一段预约的python学习之旅，共勉之！

逆水行舟用力撑，一篙松劲退千寻，古云今日足可惜，吾辈更应惜秒阴。--董必武

参考资料：

1. [Katie](http://search.dangdang.com/?key2=%C3%B7%C2%E5%C4%E1&medium=01&category_path=01.00.00.00.00.00" \o "（美）梅洛尼　著，陈宗斌　译) Cunningham. [Python入门经典](http://product.dangdang.com/23555608.html" \o " HTML与CSS入门经典(第9版)   " \t "_blank)[M]. 北京:人民邮电出版社, 2014.