

第一章 Python 3 概述

1.1 Python 简介

1.2 Python环境构建

1.3 第一个程序 Hello World !

1.4 使用 IDE PyCharm

1.5 小结

习题

Python并不是新的编程语言



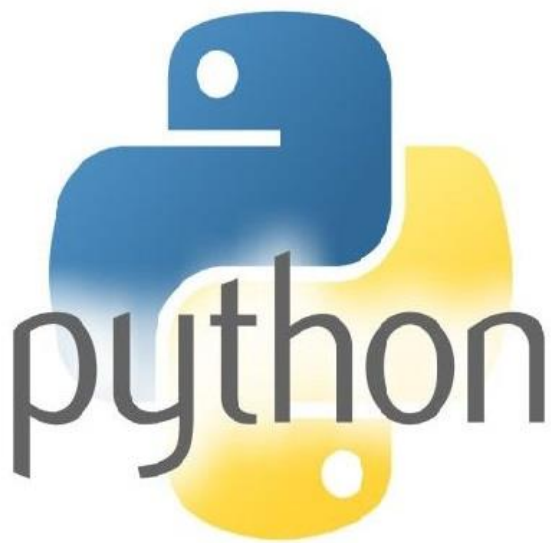
Hi Python, How old are you?

在主流编程语言当中，**Python语言**并不是一个“新人”。

Python语言目前已经超过30岁了，它早于HTTP 1.0协议5年，早于Java语言 4年。

但**Python语言**真正风靡之时却是最近几年，所以“后起之秀”的称呼实至名归。

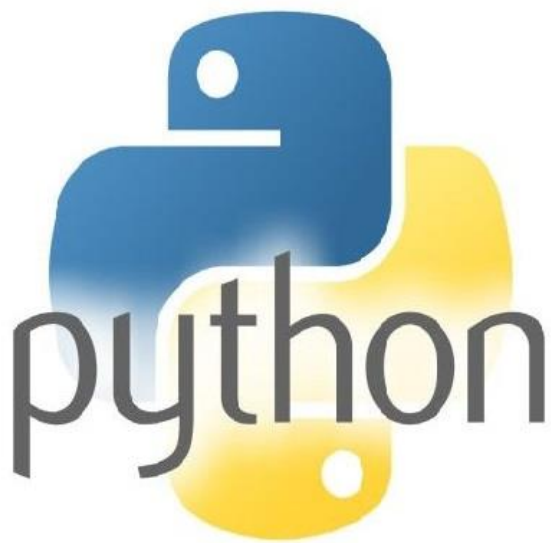
1.1.1 Python的前世今生



Python是由荷兰人Guido van Rossum（吉多·范·罗苏姆）于1989年圣诞节期间在阿姆斯特丹休假时为了打发无聊的假期而编写的一个脚本解释程序。1991年Python第一个发行第一个公开版本。

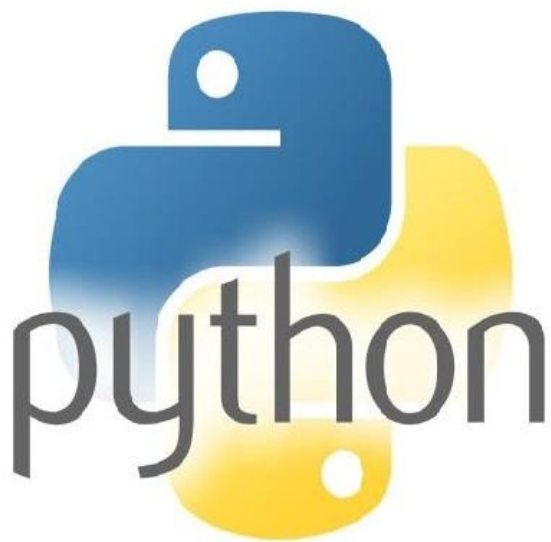
Python语言被吉多作为是ABC语言的一种继承，但坚决摒弃ABC语言的封闭性，**走开源路线**。在Python语言问世的时候，他在互联网上公开了源代码，让世界上更多喜欢Python的程序员，对Python进行不断的功能完善。这也就为后来Python的蓬勃发展奠定了坚实的基础。

1.1.1 Python的前世今生



Python能够真正风靡的原因之一是有一个好的起点。它的起步很稳，避开了版权纠纷，且搭上了开源运动的顺风车。在那个年代，商业版权一直是热门事件，业界史上第一个软件领域重大官司AT&T和伯克利BSD的Unix版权案打得天昏地暗，该案的结局直接促成了BSD的开源分支、Linux的诞生以及震惊世界的自由软件运动。

1.1.1 Python的前世今生



Python最初的版权归属是CWI（阿姆斯特丹的国家数学与计算机科研学会），这与吉多早年在该机构工作有关，后来吉多受雇于CNRI（维吉尼亚州的国家创新研究公司），Python权属转移至此。那时自由软件运动已经开始，在CNRI期间发布的1.6至2.1多个版本的Python许可证是一种与GPL并不兼容且类似于BSD的开源许可，CNRI因受到自由软件基金会的压力释放了Python的原许可证，吉多由此掌握了主导权并起草了新的许可证。他改变了原许可证与GPL的不兼容，此举获得了自由软件基金会颁发的**自由软件进步奖**。再后来吉多和他的团队成立了Python软件基金会，将版权与许可证置于其下。

1.1.1 Python的前世今生



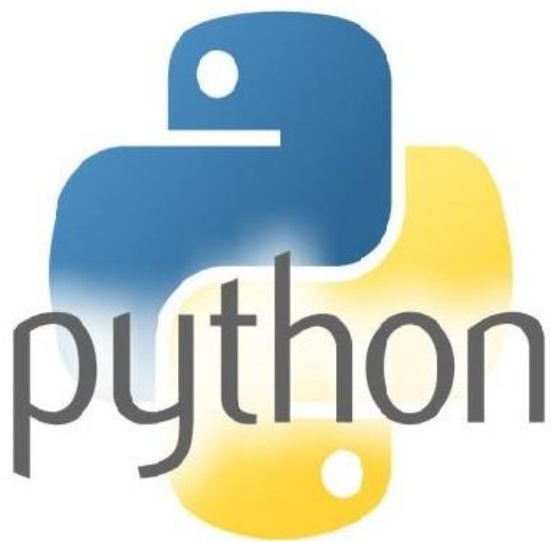
GNU通用公共许可证(GNU General Public License)简称为**GPL**，是由自由软件基金会发行的用于计算机软件的协议证书，使用该证书的软件被称为自由软件。

Linux就是采用了GPL。GPL协议和BSD，Apache License等鼓励代码重用的许可很不一样。

GPL的出发点是代码的开源/免费使用和引用/修改/衍生代码的开源/免费使用，但不允许修改后和衍生的代码做为闭源的商业软件发布和销售。

这也就是为什么我们能免费的各种Linux，包括商业公司的Linux和Linux上各种各样的由个人、组织、以及商业软件公司开发的免费软件了。

1.1.1 Python的前世今生



创始人吉多.范.罗苏姆的心思缜密与灵活处事为Python最初的发展营造了良好的环境，包括几次权属的转移、起草新的许可证、机智地与自由软件阵营斡旋，最后安全融入开源的大潮。这一切为Python此后十多年里逐渐成长为主流编程语言赢得了契机。

1.1.1 Python的前世今生

吉多·范·罗苏姆

(Guido Van Rossum, 1956年1月31日 -)

程序设计语言Python的作者，著名Python程序语言大师，被称为**Python之父**。在Python社区，吉多·范·罗苏姆被人们认为是“仁慈的独裁者

(Benevolent Dictator For Life, BDFL) ”，意思是他仍然关注Python的开发进程，并在必要的时刻做出决定。

荷兰人，1995年移居美国。荷兰阿姆斯特丹大学硕士学位。2005年至2012年底在著名IT公司Google工作。2012年底宣布加盟云存储公司Dropbox，担任软件工程师，2013年1月起在Dropbox开始工作。

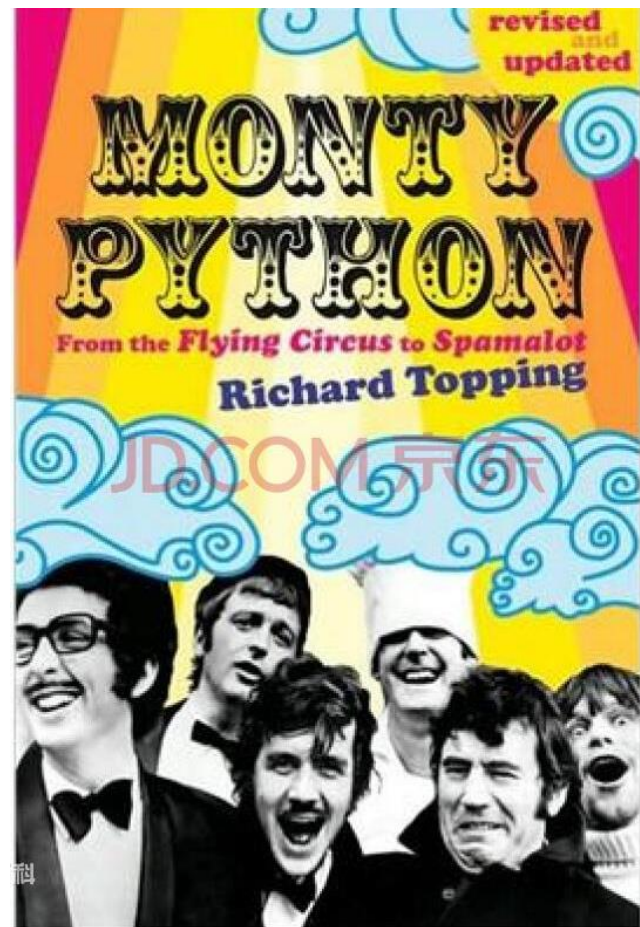


1.1.1 Python的前世今生

Python命名由来

吉多·范·罗苏姆特别喜欢电视喜剧《蒙提·派森飞行马戏团》（Monty Python's Flying Circus），因此将他创作的编程语言命名为Python。

Monty Python是英国六人喜剧团体，喜剧界的披头士。



1.1.1 Python的前世今生

现在 Python 是由一个核心开发团队在维护，吉多仍然占据着至关重要的作用，指导其进展。

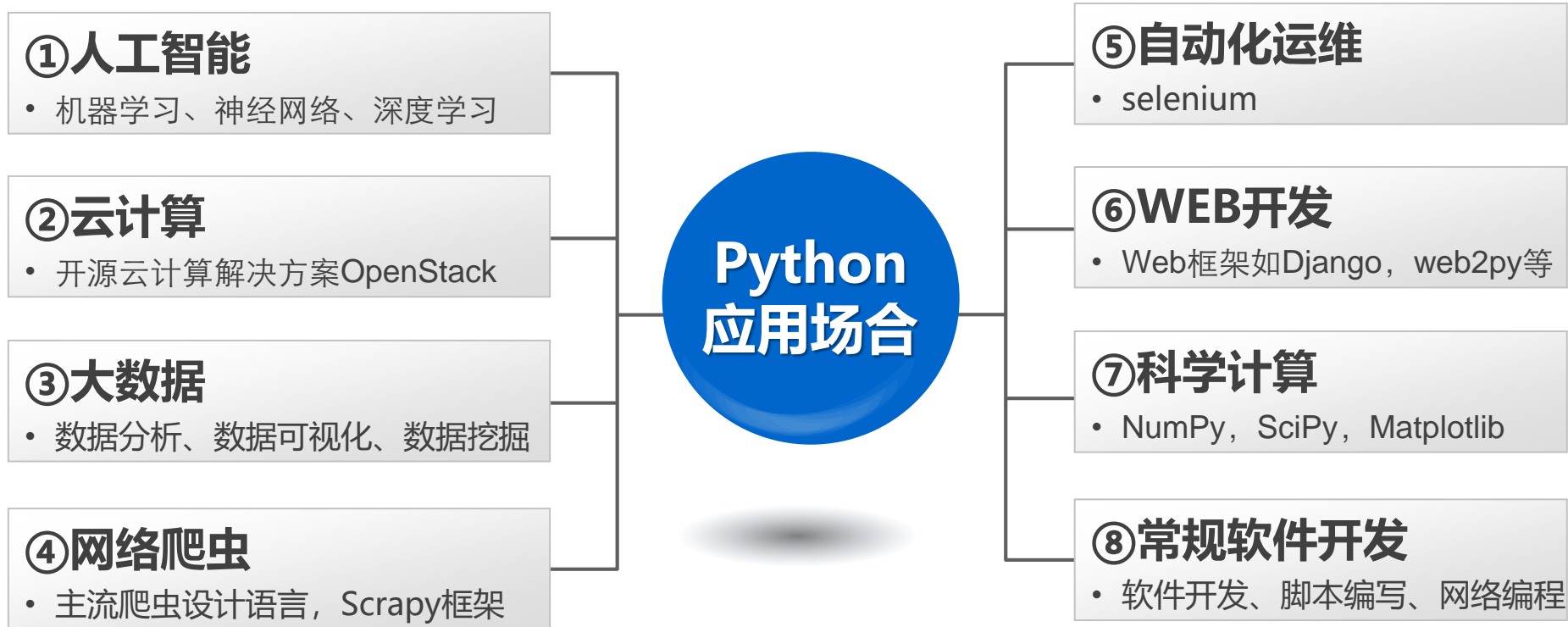
在全世界程序员不断的改进和完善下，Python现今已经成为最受欢迎的程序设计语言之一。

自从2004年以后，Python的使用率呈线性增长，2018年2月调查显示Python语言在开发语言中排名第4，仅次于Java、C和C++。**我们使用Python3.8.3版本，当下官网已经更新至3.9版本并增加了新特性。**



1.1.2 Python的应用场合

目前，Python的主要应用领域如下：



1.1.3 Python的特性

- Python简单易学
- Python是面向对象的高级语言
- Python语言是免费且开源的
- Python是解释性语言
- Python程序编写需使用规范的代码风格
- Python是可扩展和可嵌入的
- Python是可移植的
- Python运行速度快
- Python提供了丰富的库
-

1.1.3 Python的特性



Python崛起的原因之二与其本身特点有关，或者说，其长期维护演进形成的独特风格迎合了大多数开发者的口味。在开发者社群流行着一句玩笑 **“人生苦短，我用 Python”** (**Life is short, you need Python**), 这句看似戏言的话实际上恰恰反映了Python的语言特性与其在开发者心里的价值分量。

1.1.3 Python的特性

人生苦短

我用



除了包涵大多数主流编程语言的优点（面向对象、语法丰富）之外，Python的直观特点是简明优雅、易于开发，用尽量少的代码完成更多工作。

尽管Python是一种解释型语言，与传统的编译型语言相比降低了机器执行效率，但是处理器的处理速率与环境速率（比如网络环境）的差异在大多数场景中完全抵消了上述代价；牺牲部分运行效率带来的好处则是提升了开发效率，在跨平台的时候无需移植和重新编译。

所以Python的显著优点在于速成，对于时间短、变化快的需求而言尤为胜任。

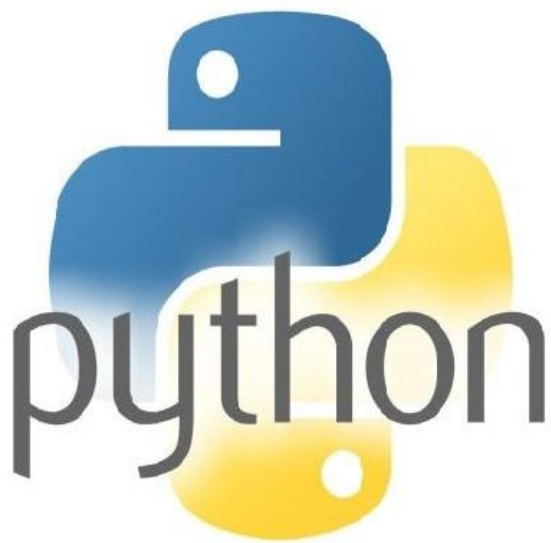
1.1.3 Python的特性



Python语法简捷和清晰，尽量使用无异义的英语单词，与其它大多数程序设计语言使用大括号不一样，它**使用缩进来定义语句块**。

```
flag = False
name = 'luren'
if name == 'python':           # 判断变量是否为 python
    flag = True                 # 条件成立时设置标志为真
    print('welcome boss')     # 并输出欢迎信息
else:
    print(name)                 # 否则输出变量名称
```


1.1.3 Python的特性

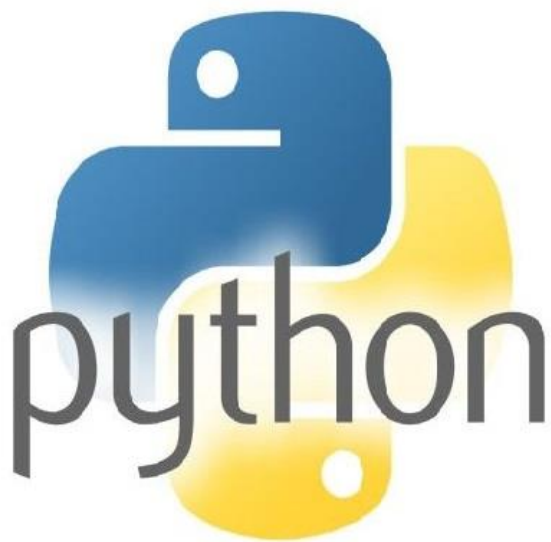


Python最强大的地方体现在它的两个外号上，一个叫“**内置电池**”，另一个是“**胶水语言**”。

前者的意思是，Python官方本身提供了非常完善的标准代码库，包括针对网络编程、输入输出、文件系统、图形处理、数据库、文本处理等等。代码库相当于已经编写完成打包供开发者使用的代码集合，程序员只需通过加载、调用等操作手段即可实现对库中函数、功能的利用，从而省去了自己编写大量代码的过程，让编程工作看起来更像是在“搭积木”。

除了内置库，开源社区和独立开发者长期为Python贡献了丰富大量的第三方库，其数量远超其他主流编程语言，可见Python的语言生态已然相当壮大。

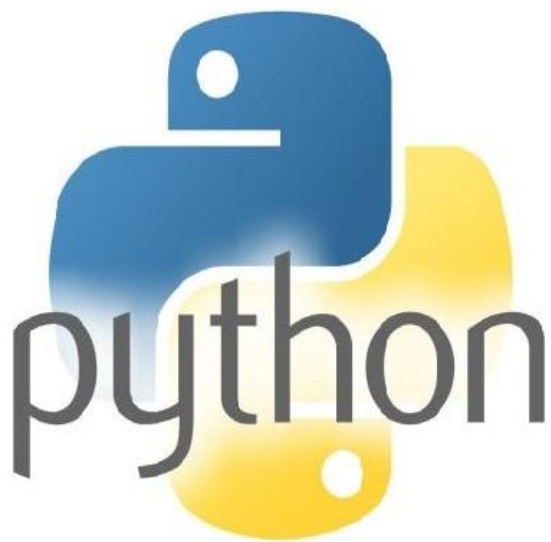
1.1.3 Python的特性



“胶水语言”是Python的另一个亮点。Python本身被设计成具有可扩展性，它提供了丰富的API和工具，以便开发者能够轻松使用包括C、C++等主流编程语言编写的模块来扩充程序。就像使用胶水一样把用其他编程语言编写的模块粘合过来，让整个程序同时兼备其他语言的优点，起到了黏合剂的作用。

正是这种多面手的角色让Python近几年在开发者世界中名声鹊起，因为互联网与移动互联时代的需求量急速倍增，大量开发者亟需一种极速、敏捷的工具来助其处理与日俱增的工作，Python发展至今的形态正好满足了他们的愿望。

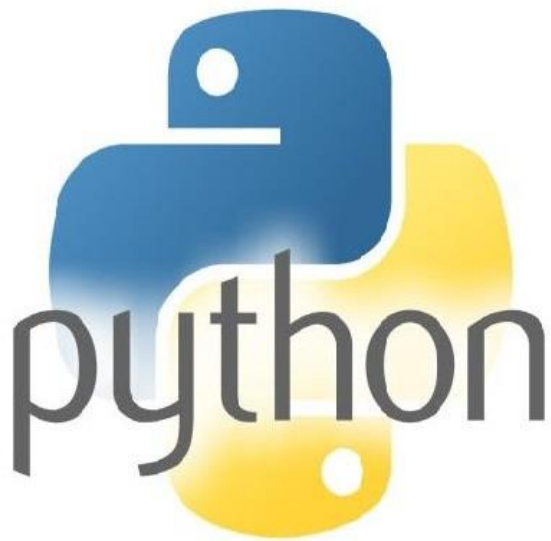
1.1.3 Python的特性



Python的应用范围，除了日常工具和脚本之外，还适用于Web程序、GUI开发、操作系统中间件、服务端运维等。这些年Python的一些第三方库在机器学习、神经网络方面活跃非凡，这也为语言本身的推广和流行加分不少。

在国内，很多大家比较熟悉的网站都是用python开发的，比如豆瓣、知乎、网易、百度、阿里、土豆、新浪等;国外的话，谷歌、YouTube、Facebook等企业也在广泛使用python。这说明Python语言本身的发展已日臻完善，有着极高的稳定与可靠性保证。

1.1.3 Python的特性



Python编程思想**包含强烈的黑箱思维**，这意味着开发者将愈加重视模块化和流水线式的编程工作，事实上这也是未来主流编程语言的发展趋向。随着计算机语言的演化和开发工具集成功能日趋强大，未来的编程工作将大幅简化。

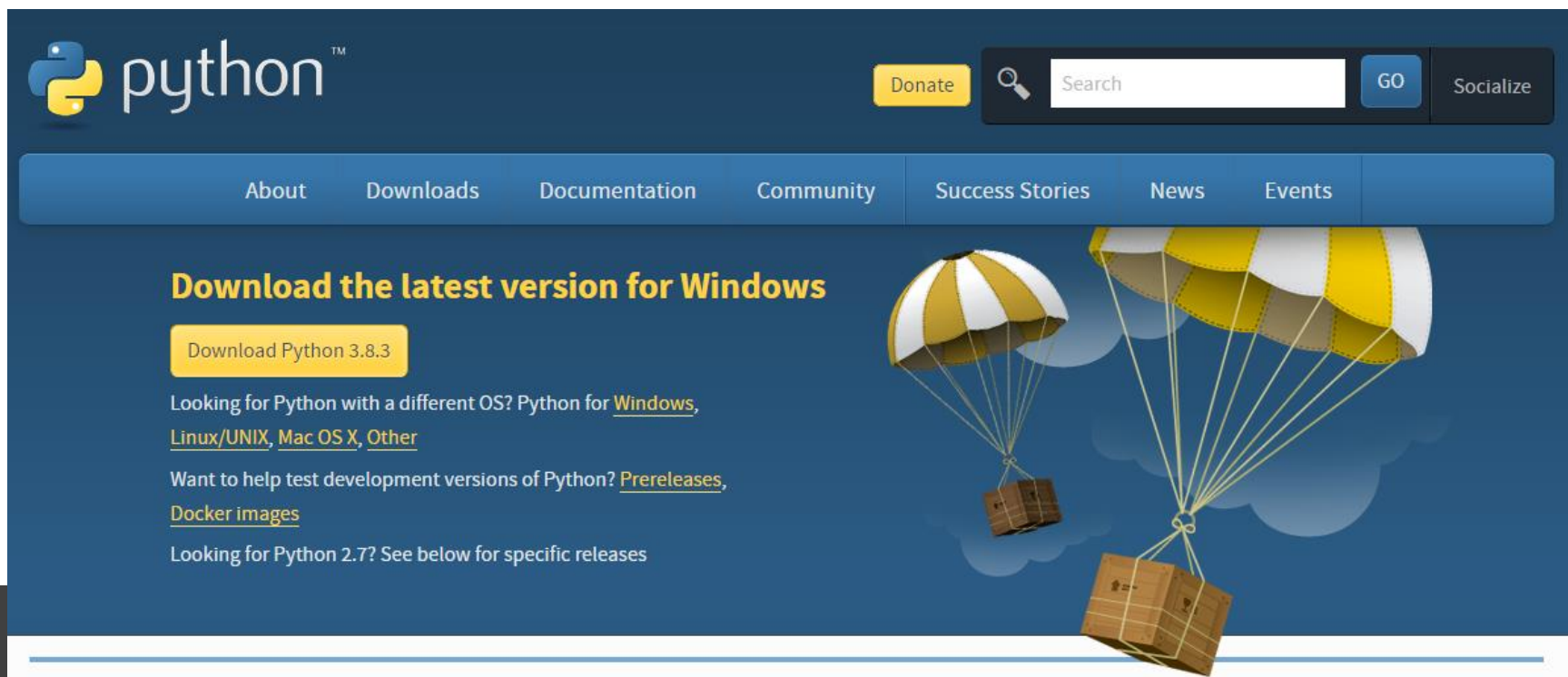
从某种角度看，Python更像是已经“迈入未来”的编程语言，其对开发者群体结构变化，以及新进开发者数量的激增，这些影响都将是深远的。

1.1.4 选择Python的版本

2008年10月Python 3.0版本发布，该版本在Python 2的上进行了很大的改变，使得两者互不兼容。我们该如何选择Python的版本呢？

由于Python 3相较于Python 2还有大量的改进和提升的地方，这就使得Python 2有了些许“鸡肋”之感。

因此，我们跟随技术的发展和前进的潮流，选择Python 3作为我们学习的对象。



1.1.5 如何学习Python

- **基础语法**：先学习、后模仿、再自主创新。了解Python的数据类型、变量、判断、循环、函数、类等等，逐步找到编程的感觉。
- **积极实践**：俗话说“拳不离手，曲不离口”，程序编写水平是在不断的练习和实践中提高的。
- **遵守规范**：建议使用Python编程的开发者，都应遵循PEP8规范
- **自主学习**：也许你为了完成某些特定功能，需要使用一些还未了解的技术，那就不要犹豫和等待，DIY！
- **善于交流**：积极主动的和其他学习者交流，取长补短。

Python3的新特性

- `print()` and `exec()` 函数, 旧版本里, `print`和`exec`是作为一语句使用:
`print "Hello,World!"` ,
在3.0中,应写成:
`print ("Hello,World!")`。

- 用`input()`代替 `raw_input()`,
- 源文件编码从ASCII变为UTF-8。
- 比较: Python3对于值的比较要严格得多。在Python2中, 任意两个对象均可进行比较。

- 异常处理如:
 - * 增加异常基 `BaseException` ;
 - * 移除了 `StandardError`;
 - * 抛出异常: 使用 `raise Exception(args)`
 - * 捕获异常: 使用 `except Exception as identifier`
 - * 异常链 (Exception chain)

1

2

3

4

5

6

- `int`和`long`统一为`int`, `int`表示任何精度的整数, 移除 `sys.maxint`, 因为`int`已经是最大的整数。新版本移除了含糊的除法符号 `'/'` , 而只返回浮点数。

- 标识符支持非ASCII 字符, 例如:
姓名 = "张三"
`class` 男人:
 `@classmethod`
 汤姆 = 男人()

- 字符串格式化变化: 格式化字符串内置的`%`操作符太有限了, 新版本新增加了`format()`, 比以前更灵活了, `%`要逐渐被淘汰。

第一章 Python3概述

1.1 Python 简介

1.2 Python环境构建

1.3 第一个程序 Hello World !

1.4 使用 IDE PyCharm

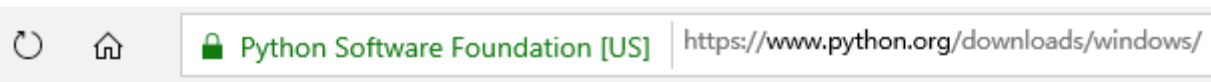
1.5 小结

习题

1.2.1 在Windows系统中安装Python 3

进入Python官方网站 (<https://www.Python.org>) 下载安装包

进入Windows版的下载页面 (<https://www.Python.org/downloads/windows/>)



Stable Releases

▪ [Python 3.8.3 - May 13, 2020](#)

Note that Python 3.8.3 *cannot* be used on Windows XP or earlier.

- Download [Windows help file](#)
- Download [Windows x86-64 embeddable zip file](#)
- Download [Windows x86-64 executable installer](#)
- Download [Windows x86-64 web-based installer](#)
- Download [Windows x86 embeddable zip file](#)
- Download [Windows x86 executable installer](#)
- Download [Windows x86 web-based installer](#)

web-based installer: 基于web的安装文件，安装过程中需要一直连接网络；

executable installer: 是可执行的安装文件，下载后直接双击开始安装；

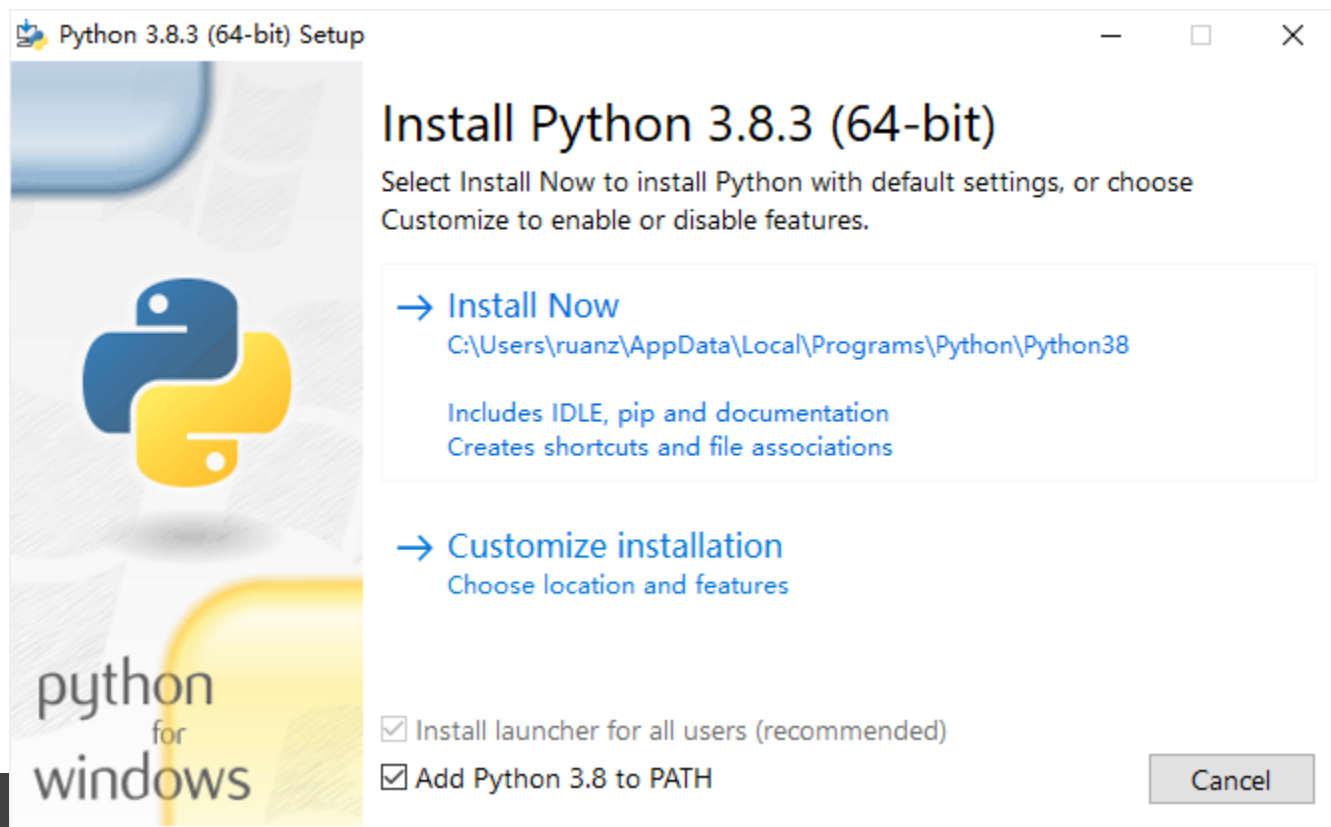
embeddable zip file: 是安装文件的zip格式压缩包，下载后需要解压缩之后再进行安装。

Windows x86-64 executable installer: x86架构的计算机的windows 64位操作系统的可执行安装文件。

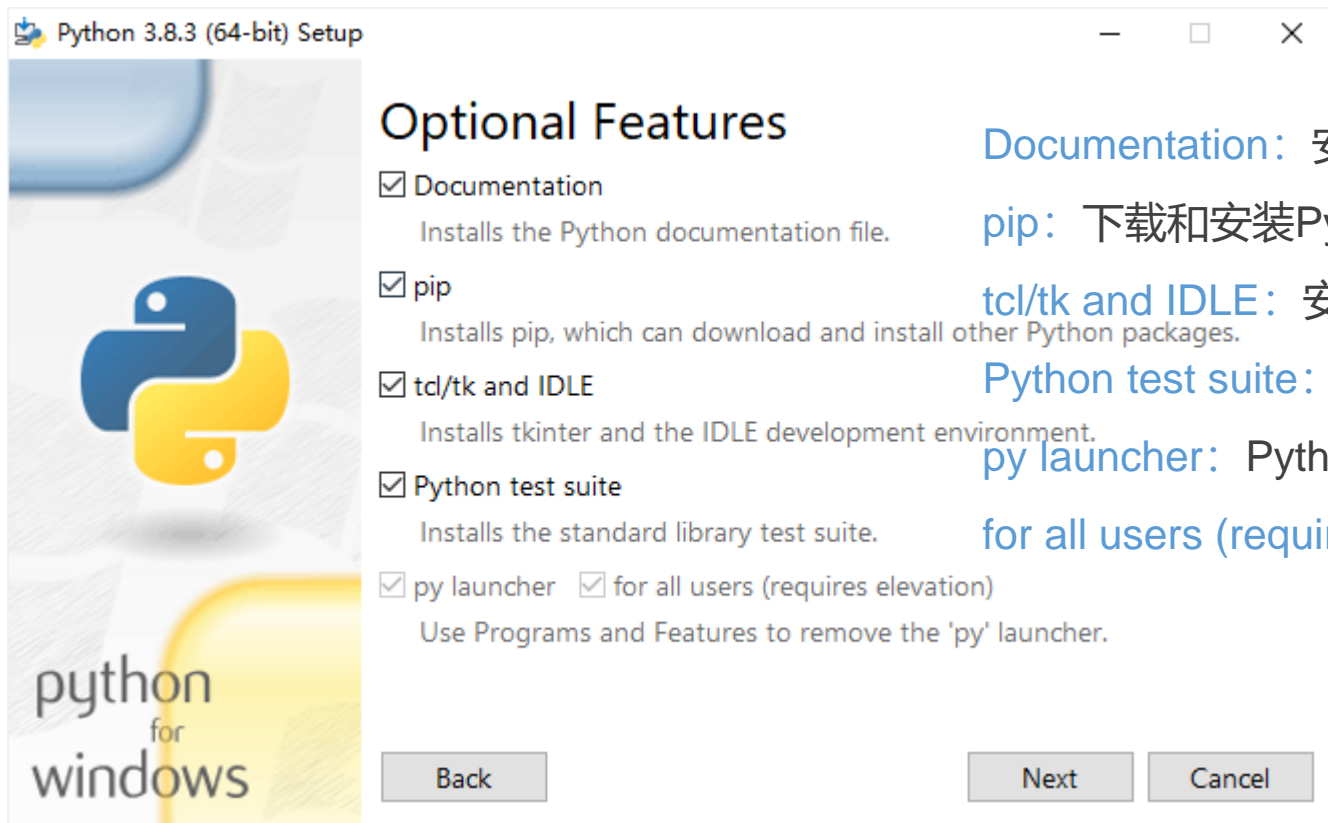
这里我们下载的是 “Windows x86-64 executable installer”

1.2.1 在Windows系统中安装Python 3

安装时需要选中最下方的Add Python 3.8 to PATH，即把Python3.8的可执行文件、库文件等路径，添加到环境变量，这样可以在windows shell环境下面运行Python。



1.2.1 在Windows系统中安装Python 3



Documentation: 安装Python文档文件

pip: 下载和安装Python包的工具

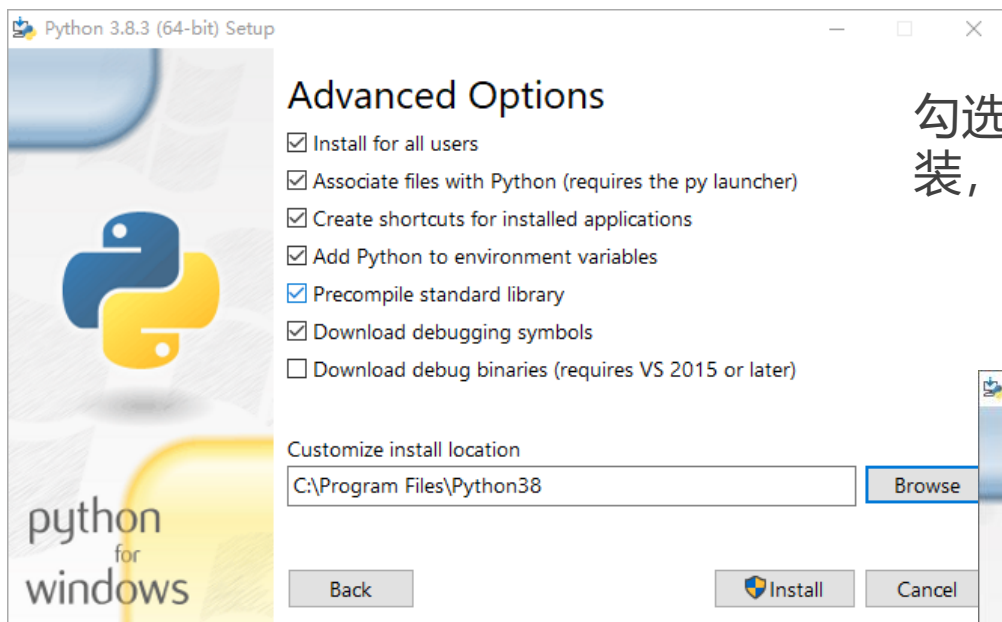
tcl/tk and IDLE: 安装tkinter和IDLE开发环境

Python test suite: Python标准库测试套件

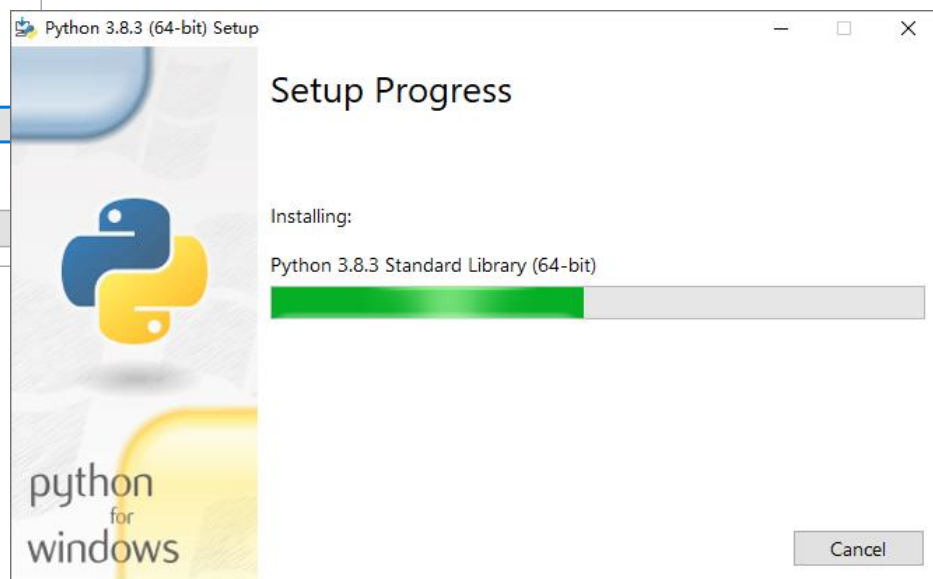
py launcher: Python启动器

for all users (requires elevation): 所有用户使用

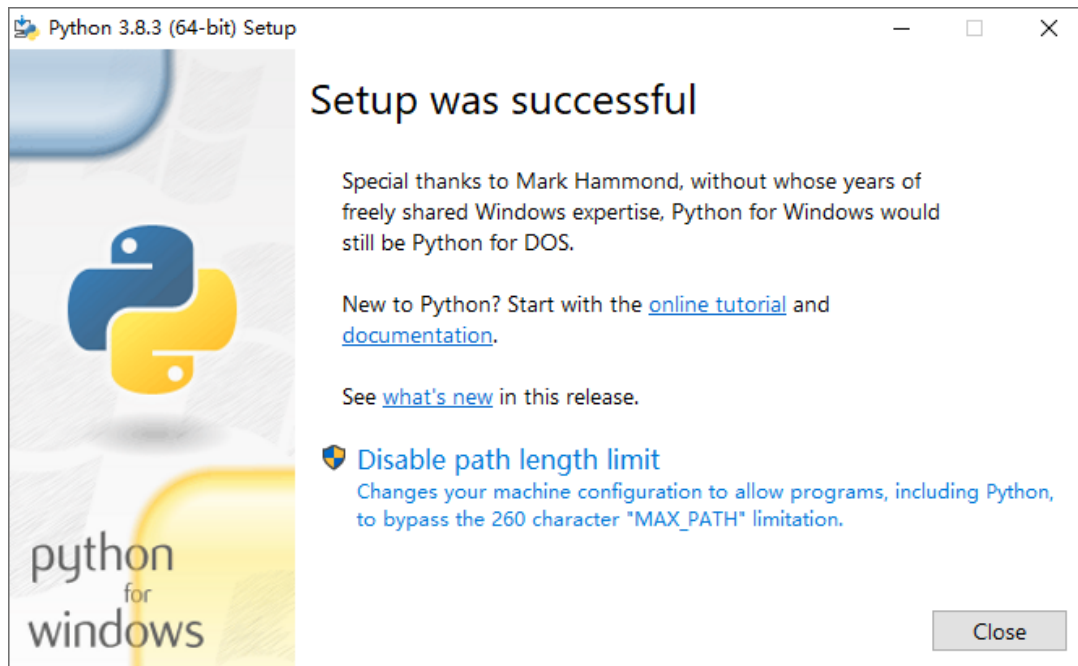
1.2.1 在Windows系统中安装Python 3



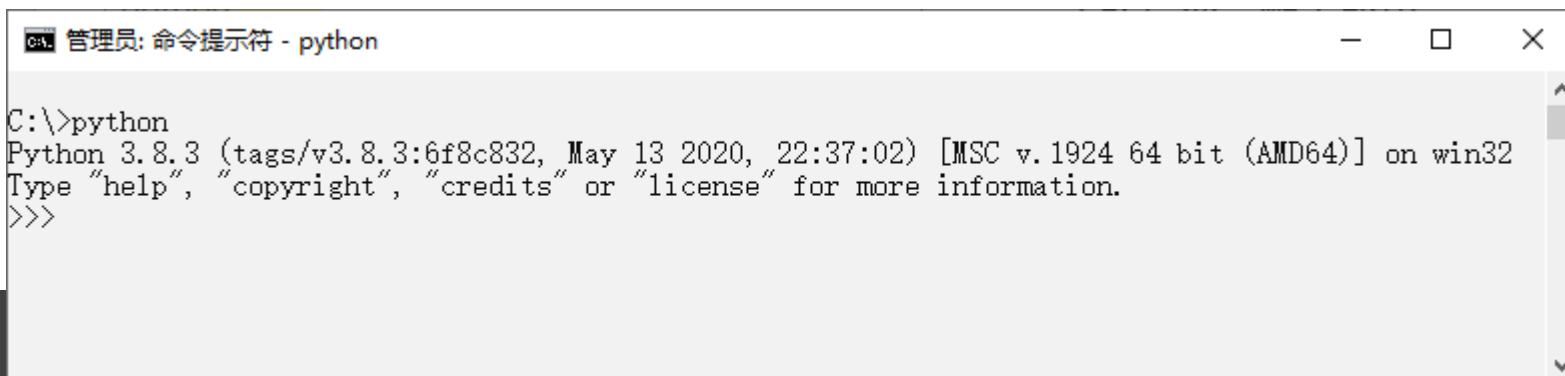
勾选Install for all users针对所有用户安装，就可以按自己的需求修改安装路径



1.2.1 在Windows系统中安装Python 3



安装完成
使用命令提示符进行验证，
打开Windows的命令行模式，
输入 “ Python ” 或
“python”，屏幕输出如下图
所示，则说明Python解释器成
功运行，Python安装完成，并
且相关环境变量配置成功。



1.2.2 在Linux系统中安装Python 3

Linux系统中自带安装有Python 2.7，我们建议不要去改动它，因为系统中有依赖目前的Python 2的程序。

本部分使用Ubuntu 18.04进行安装示例，其他发行版的安装方法，见Python官网的说明。

1.2.2 在Linux系统中安装Python 3

更新软件包列表并安装构建Python所需的软件包：

```
ruanzl@small:~$ sudo apt update
```

```
ruanzl@small:~$ sudo apt install build-essential zlib1g-dev libncurses5-d  
ev libgdbm-dev libnss3-dev libssl-dev libreadline-dev libffi-dev wget
```

再使用wget命令到Python官网下载3.8.3的源码安装包：

```
ruanzl@small:~$ wget https://www.python.org/ftp/python/3.8.3/Python-  
3.8.3.tgz
```

下载完成后使用tar命令对压缩包解压：

```
ruanzl@small:~$ tar -xf Python-3.8.3.tgz
```

1.2.2 在Linux系统中安装Python 3

解压完成后会在当前目录下生产目录python-3.8.3，使用cd命令切换到该目录下，编译安装：

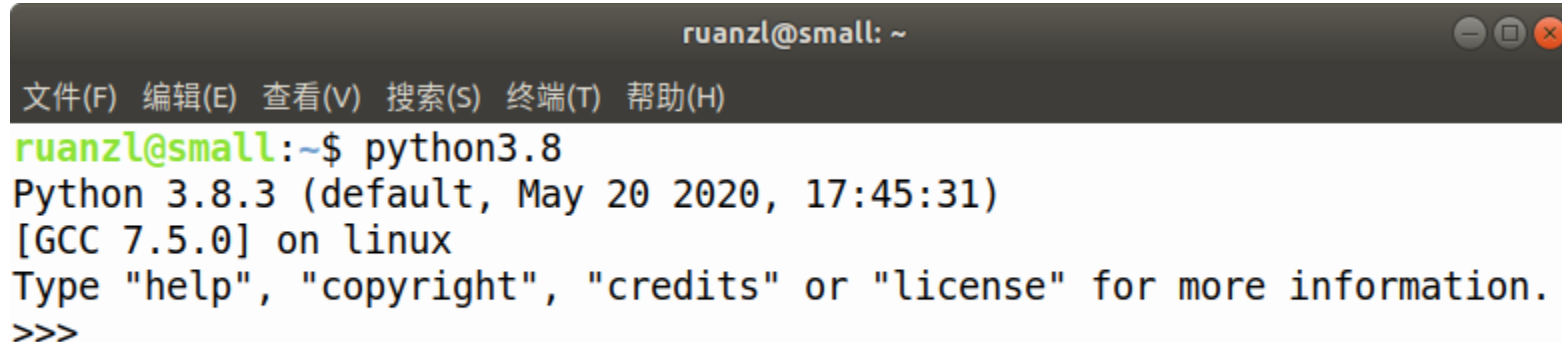
```
ruanzl@small:~$ cd Python-3.8.3
```

```
ruanzl@small:~/Python-3.8.3$ ./configure # 执行configure脚本，该脚本执行许多检查以确保系统上的所有依赖项都存在
```

```
ruanzl@small:~/Python-3.8.3$ sudo make # 编译构建python3
```

```
ruanzl@small:~/Python-3.8.3$ sudo make install # 安装python3
```

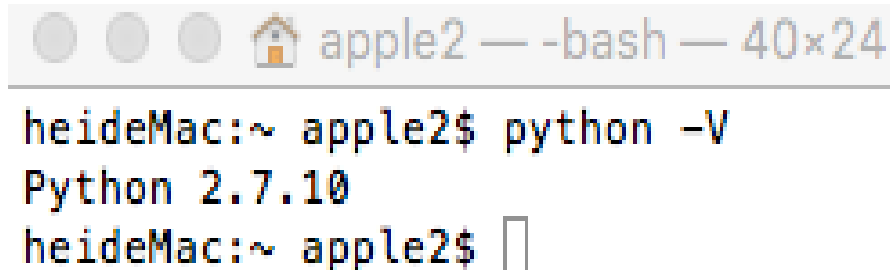
编译安装完成后，在命令行使用 `python3.8` 命令运行Python3的解释器，验证安装，如下图所示



```
ruanzl@small: ~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
ruanzl@small:~$ python3.8  
Python 3.8.3 (default, May 20 2020, 17:45:31)  
[GCC 7.5.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>>
```

1.2.3 在Mac OS系统中安装Python 3

Mac上自带的是Python版本2.X, 如果需要Python 3.X, 则需要自己手动进行安装可以使用 `python -V` 在终端查看自己的Python版本。



```
apple2 — -bash — 40x24
heideMac:~ apple2$ python -V
Python 2.7.10
heideMac:~ apple2$
```


1.2.3 在Mac OS系统中安装Python 3

访问 Python官方的下载页面

<https://www.python.org/downloads/mac-osx/>

下载Mac版本的Python 3.8.3



1.2.3 在Mac OS系统中安装Python 3

下载完成后，双击安装文件，一直点击继续进行安装，安装过程较简单，安装完成后，在Launchpad中会多了两个APP，如下图所示。



点击IDLE进入Python解释器的shell，便可验证安装。

第一章 Python3概述

1.1 Python 简介

1.2 Python环境构建

1.3 第一个程序 Hello World !

1.4 使用 IDE PyCharm

1.5 小结

习题

Python 3.8.3安装完成后，其自有的集成开发和学习环境IDLE (Python's Integrated Development and Learning Environment) 就可以开始进行编程了。

编写并执行我们的第一个Python程序 “Hello World!”

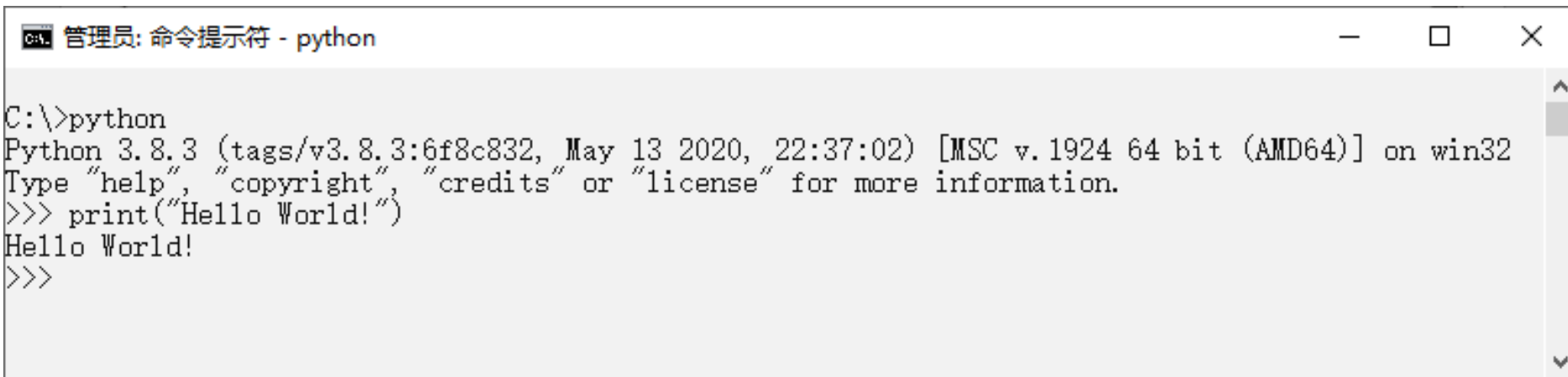
```
print("Hello World!")
```

第一种方法，在命令行模式下，进入Python解释器进行代码编写，该方法可以简单快速的开始我们的编程。

在Windows (Windows 7或10) 操作系统中，使用快捷键“win” + “R”，弹出“运行”窗口，输入cmd并确定，输入Python进入Python命令行，在提示符“>>>”之后，可以输入程序代码。这里输入第一个Python程序的代码：

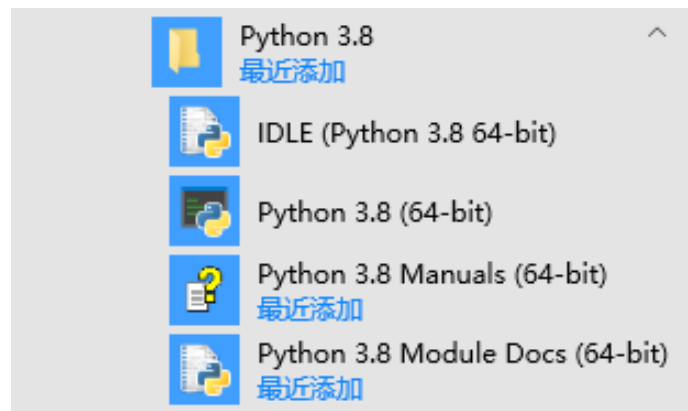
```
>>> print("Hello World!")
```

完成输入后回车执行，执行结果显示在该代码下一行，如下所示：



```
C:\>python
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World!")
Hello World!
>>>
```

第二种方法，点击Windows的“开始”菜单，从程序组中找到“Python 3.8”下的“IDLE (Python 3.8 64-bit)”快捷方式，如右图所示：



点击并进入到Python IDLE Shell窗口，在提示符“>>>”之后，输入第一个Python程序的代码：

```
>>> print("Hello World!")
```

完成输入后回车执行，如下图所示：

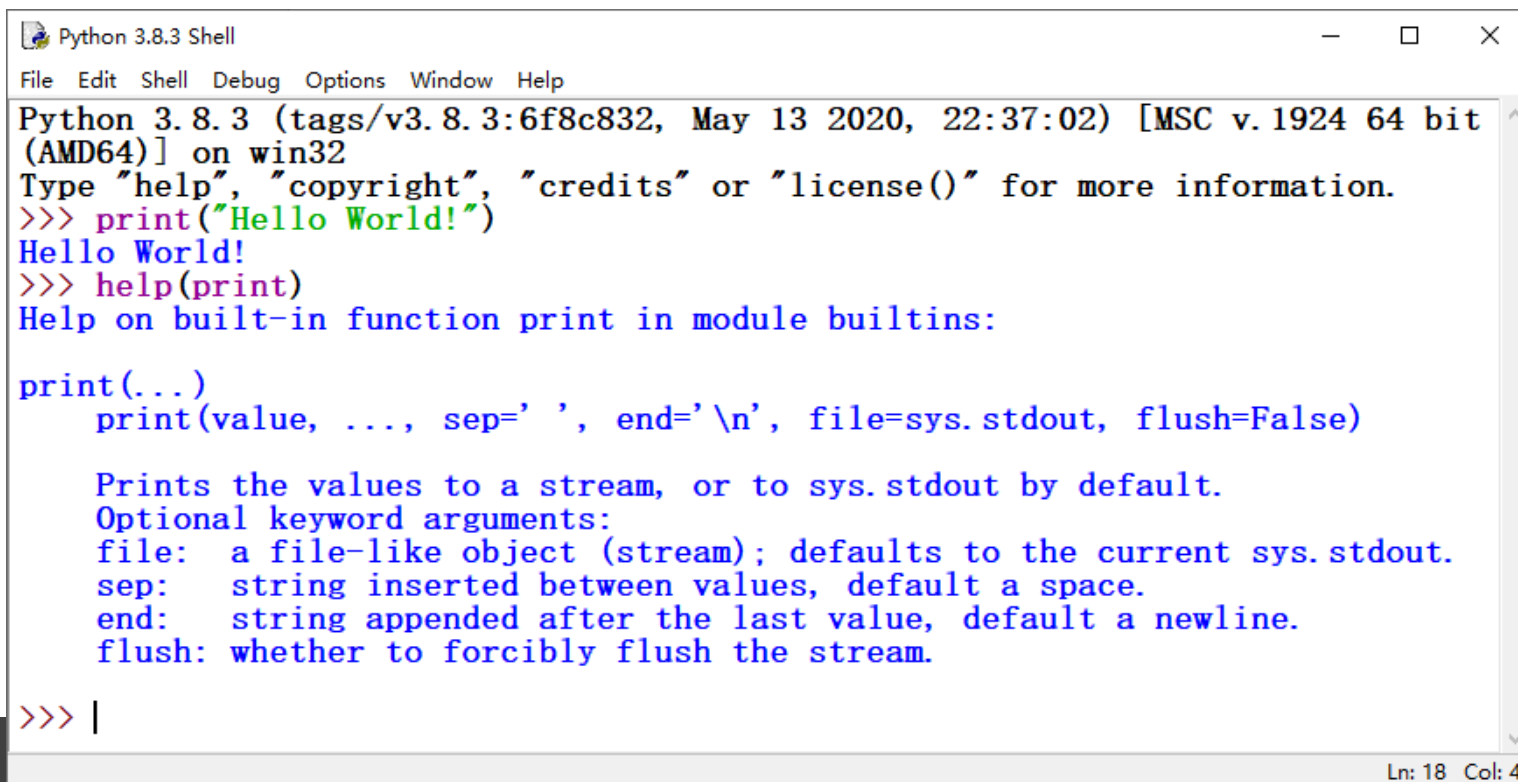
```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello World!")
Hello World!
>>> |
```

1.3.1 程序简析

`print("Hello World!")`

`print()`: Python内置函数(built-in function)名称, 作用是输出括号中的内容;

"Hello World": 字符串类型的数据, 作为参数传递给`print`函数



```
Python 3.8.3 Shell
File Edit Shell Debug Options Window Help
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello World!")
Hello World!
>>> help(print)
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file: a file-like object (stream); defaults to the current sys.stdout.
    sep: string inserted between values, default a space.
    end: string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.

>>> |
```

Ln: 18 Col: 4

1.3.2 print()函数

print()函数做进一步说明，它的基本用法是：

print("参数")

print()是函数，参数就是需要输出的内容，这些内容可以是**数值、字符串、布尔、列表或字典**等数据类型。

如果要输出多个参数，参数与参数之间用逗号隔开，如：

print("China", countries)

双引号（或者使用单引号）内的内容称为字符串常量，照原样输出内容；没有引号的countries是变量，会输出代表内容；print()函数执行完成后**默认换行**，如不需要换行，则在输出内容之后加上end = ""，如：

print(i,end='')

print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

1.3.3 input()函数

input()函数是Python语言中值的最基本输入方法，通过用户输入，接受一个标准输入数据，它从命令行读取一行，默认为 string 类型，基本用法：

object = input('提示信息')

object是需要接收用户输入的对象，提示信息的内容在函数执行时会显示在屏幕上，用于提示用户输入。提示信息可以为空，即括号内无内容，函数执行时不会提示信息。

```
>>> proverb = input("请输入一条谚语:")      #定义变量
请输入一条谚语: No pain, no gain!           #执行，输入字符串
>>> print(proverb)
No pain, no gain!                           #返回结果
```

1.3.3 input()函数

Input()函数的数据输入时默认为字符串类型，可以使用数据类型转换函数进行转换，如：

```
>>> age = input("请输入年龄:")    #定义变量
请输入年龄:18                      #执行，输入数值
>>> print(type(age))               #查看变量类型
<class 'str'>                      #返回结果

>>> age = int(input("请输入年龄:")) #重置变量，嵌套整型转换
请输入年龄:18                      #执行，输入数值
>>> print(type(age))               #查看变量类型
<class 'int'>                     #返回结果
```

- Python2中还有raw_input([prompt]) 函数，用于从标准输入读取一个行，并返回一个字符串（去掉结尾的换行符）。
- 但是，Python3中取消了该函数。

1.3.4 注释

在Python代码中加入必要的注释，使其具有较好的可读性。

注释分为两种，单行注释和多行注释。

单行注释：使用 “#” ，其后（右边）的内容将不会被执行，例如：

单行注释的内容

单行注释一般可放在一行程序代码之后，或者独自成行。

多行注释：使用两组，每组三个连续的双引号（或者单引号），两组引号之间为多行注释的内容，例如：

"""

多行注释的内容

"""

一个标准的完整的Python程序文件的头部，应有相关注释来记录编写者姓名，实现的功能和编写日期（修改日期）等重要信息。

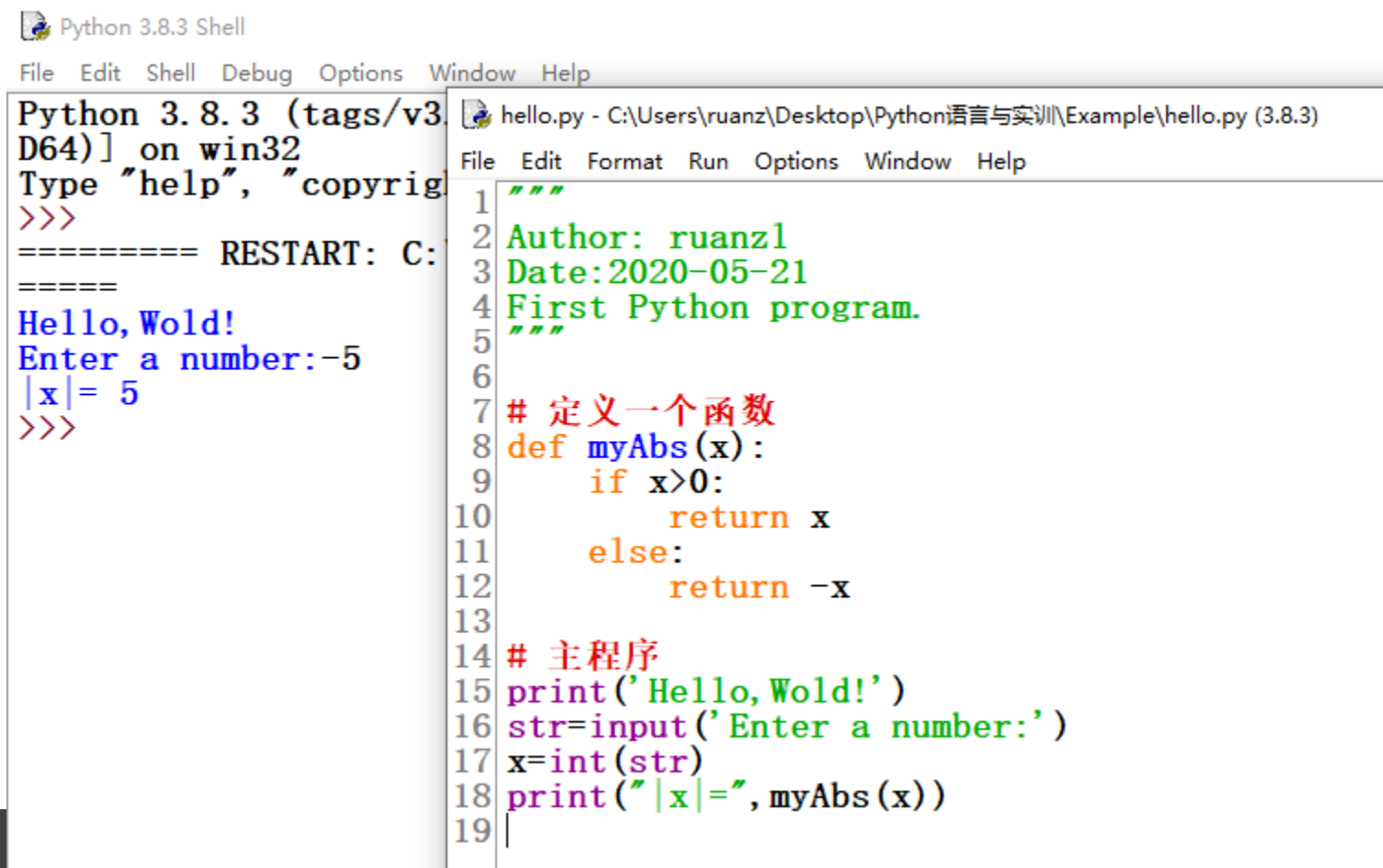
1.3.5 IDLE使用简介

为了更好地使用IDLE进行Python程序编写，这里介绍一下IDLE的使用方法。

IDLE作为Python的默认开发和学习工具，具有以下特点：

1. IDLE是一个百分百的纯Python编写的应用程序，使用了tkinter的用户界面工具集 (tkinter GUI toolkit) 。
2. 跨平台，在Windows、Unix和Mac OS X上具有相同的效果。
3. **交互式的解释器，对代码的输入、运行结果的输出和错误信息均有友好的颜色提示，并且用户可自定义显示的颜色方案。**
4. **支持多窗口的代码编辑器，也支持多重撤销，Python语法颜色区分，智能缩进，调用提示和自动补全等。**
5. 任意窗口内的搜索，编辑器窗口中的替换，以及多文件中的查找。
6. **具有断点，步进，及全局和本地命名空间的调试器。**

创建Python脚本文件并运行



1.3.6 一行多语句和一语句多行

Python3 中，一行可以书写多个语句，一个语句也可以分成多行书写。

- 一行可以书写多个语句，语句之间用分号隔开即可，例如：

```
print('I love you'); print('very much!')
```

- 一行过长的语句可以使用反斜杠或者括号分解成几行，例如：

```
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0): # 闰年条件
```



```
if (year % 4 == 0 and year % 100 != 0) \  
    or (year % 400 == 0): # 闰年条件
```



```
if ((year % 4 == 0 and year % 100 != 0) \  
    or (year % 400 == 0)): # 闰年条件
```

第一章 Python3概述

1.1 Python 简介

1.2 Python环境构建

1.3 第一个程序 Hello World !

1.4 使用 IDE PyCharm

1.5 小结

习题

1.4.1 PyCharm的安装

PyCharm 是JetBrains推出的一款Python的集成开发环境（IDE），具备一般 IDE 的常用功能，比如：调试、语法高亮显示、项目管理、代码跳转、智能提示、自动完成和版本控制等。另外，PyCharm 还提供了一些用于 Django（一种基于Python的web应用框架）开发的功能，同时支持 Google App Engine和 IronPython。

PyCharm有两个重要版本：社区版和专业版；其中社区版是免费提供给使用者学习Python的版本，其功能可以满足我们目前的学习需求

官方下载地址：<http://www.jetbrains.com/pycharm/download/>

1.4.1 PyCharm的安装

在官方网站下载最新版本的PyCharm社区版,
<https://www.jetbrains.com/pycharm/download/#section=windows>



Version: 2020.1.1
Build: 201.7223.92
7 May 2020

[System requirements](#)
[Installation Instructions](#)
[Other versions](#)

Download PyCharm

[Windows](#) [Mac](#) [Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free trial

Community

For pure Python development

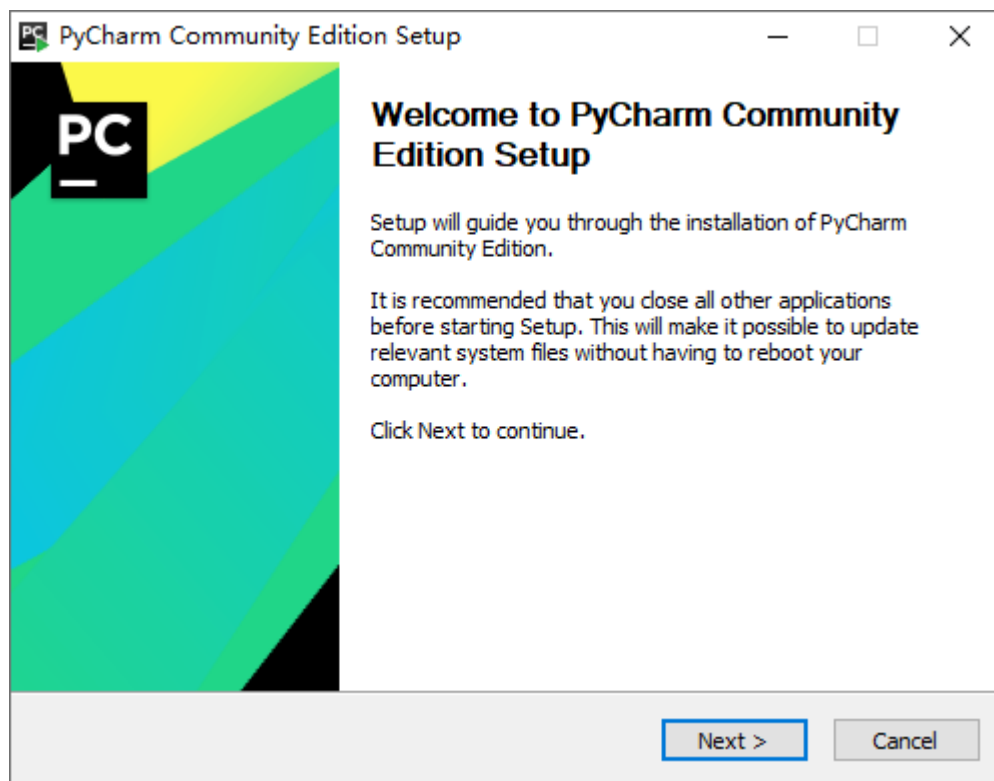
Download

Free, open-source

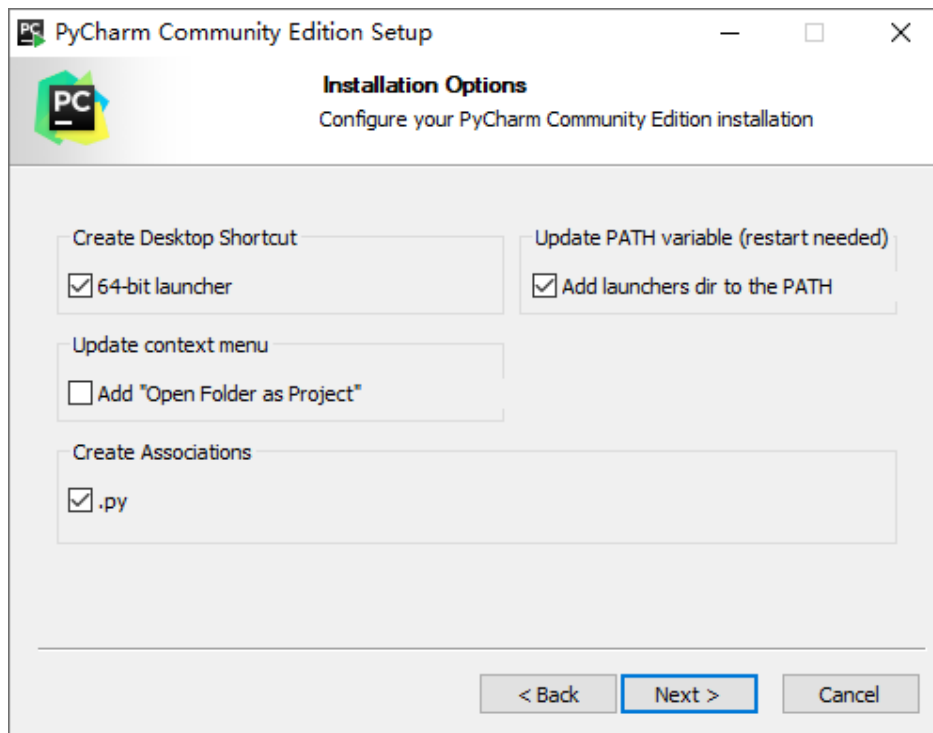
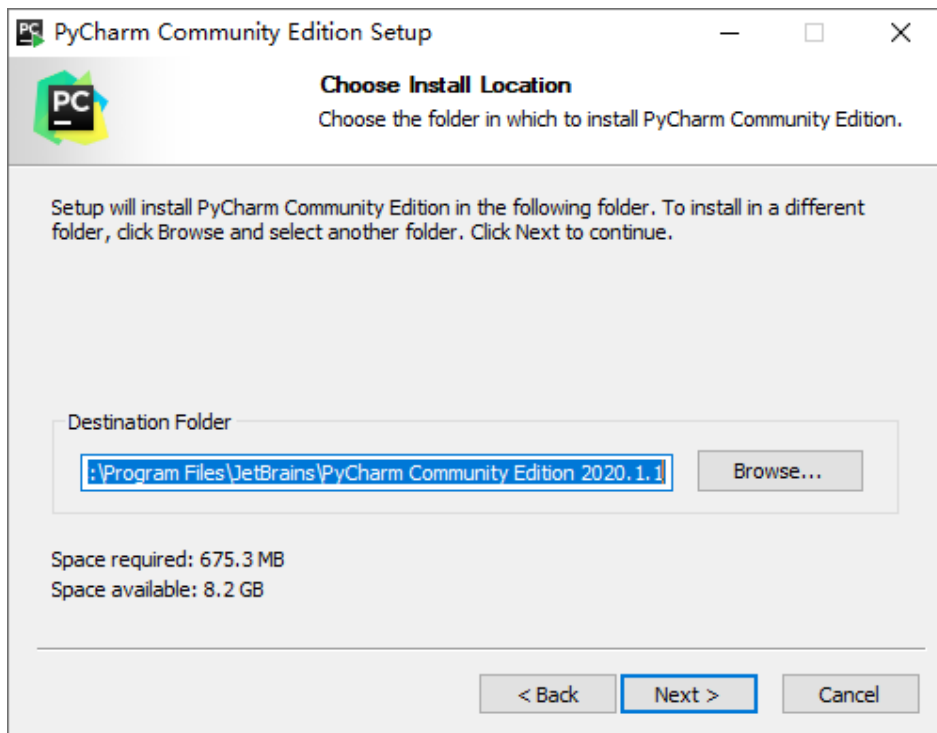
下载页面提供了适用于Windows、macOS和Linux等操作系统的各版本PyCharm下载，其中Professional（专业版）可供试用，Community（社区版）是轻量级（Lightweight）的免费版，这里点击Community（社区版）下的DOWNLOAD下载该版本。

1.4.1 PyCharm的安装

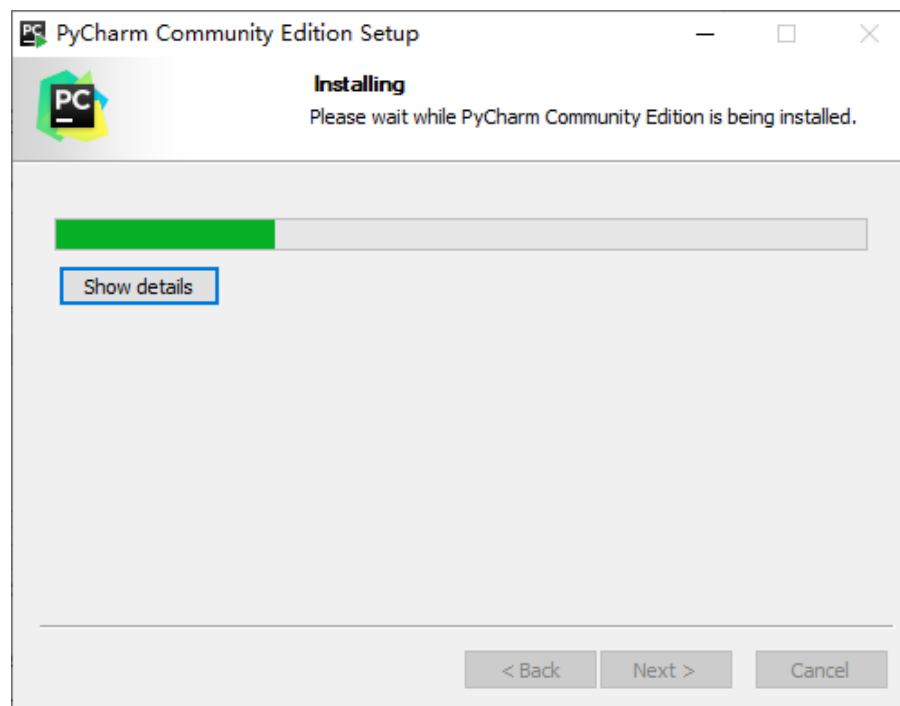
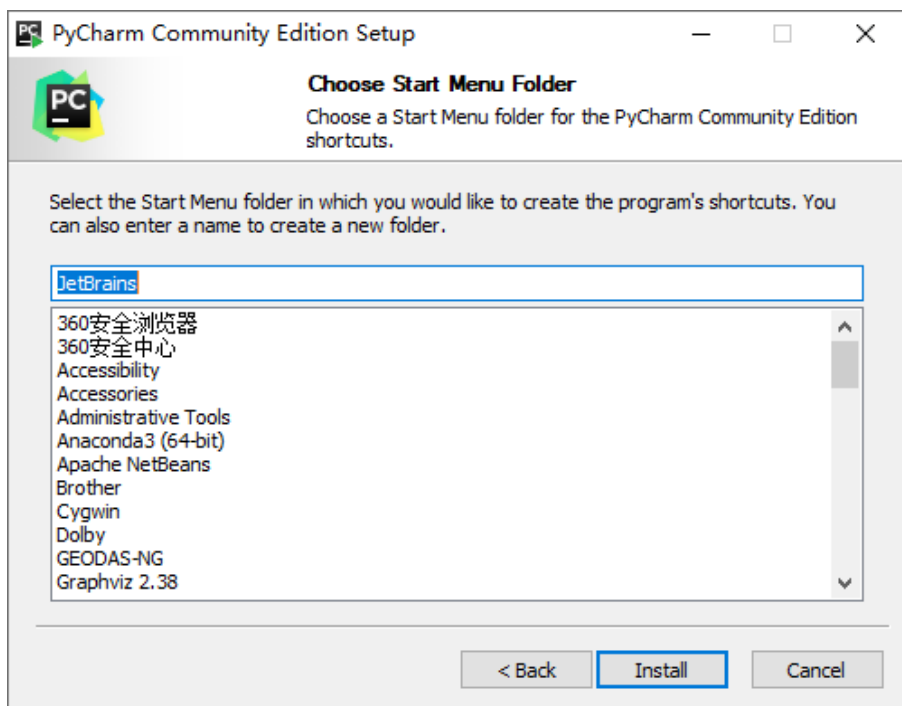
下载完成后双击进入安装向导，如下图所示：



1.4.1 PyCharm的安装

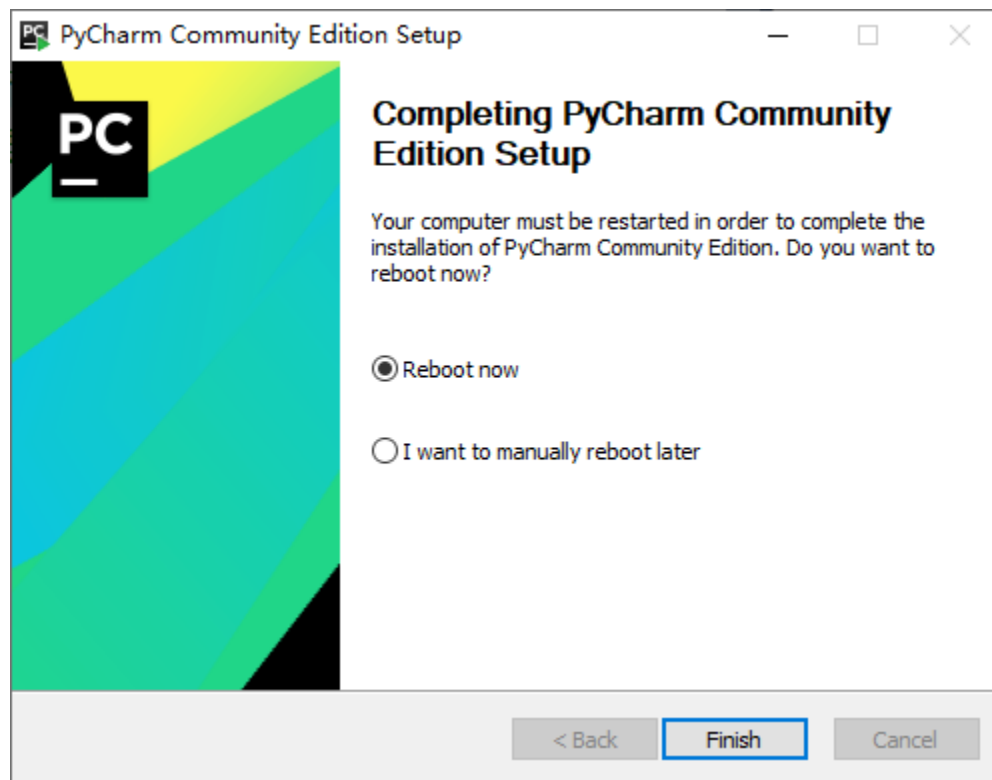


1.4.1 PyCharm的安装



点击Install开始安装

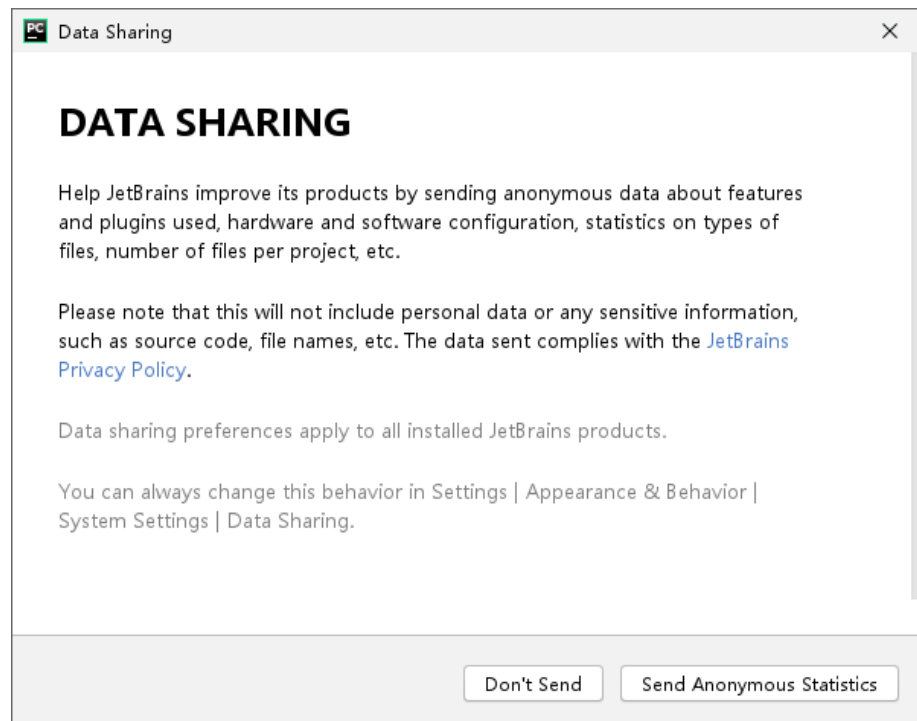
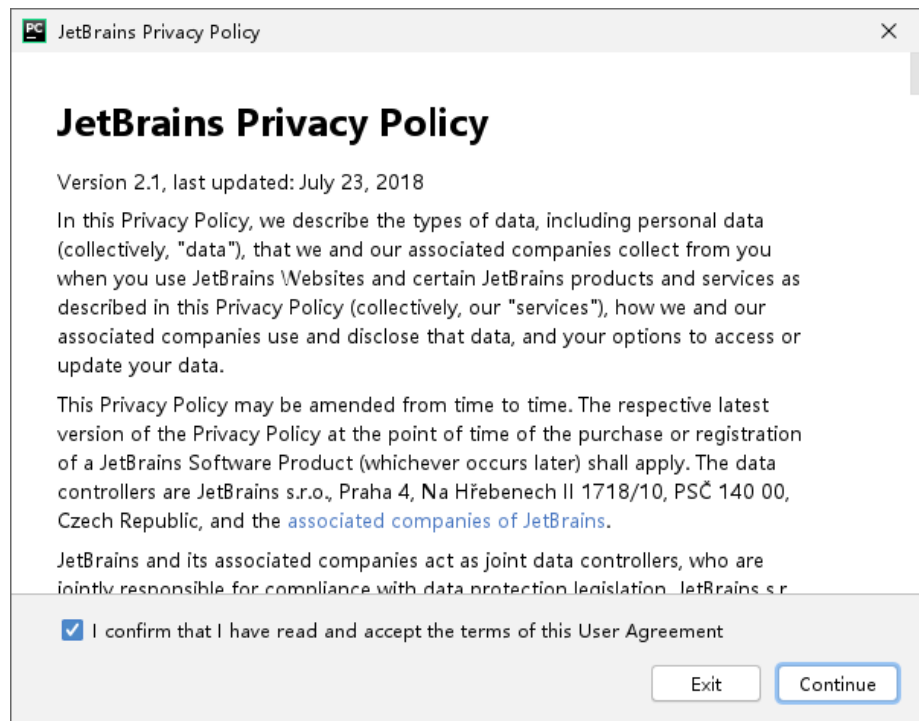
1.4.1 PyCharm的安装



安装完成，点击Finish重启计算机

1.4.2 PyCharm的启动

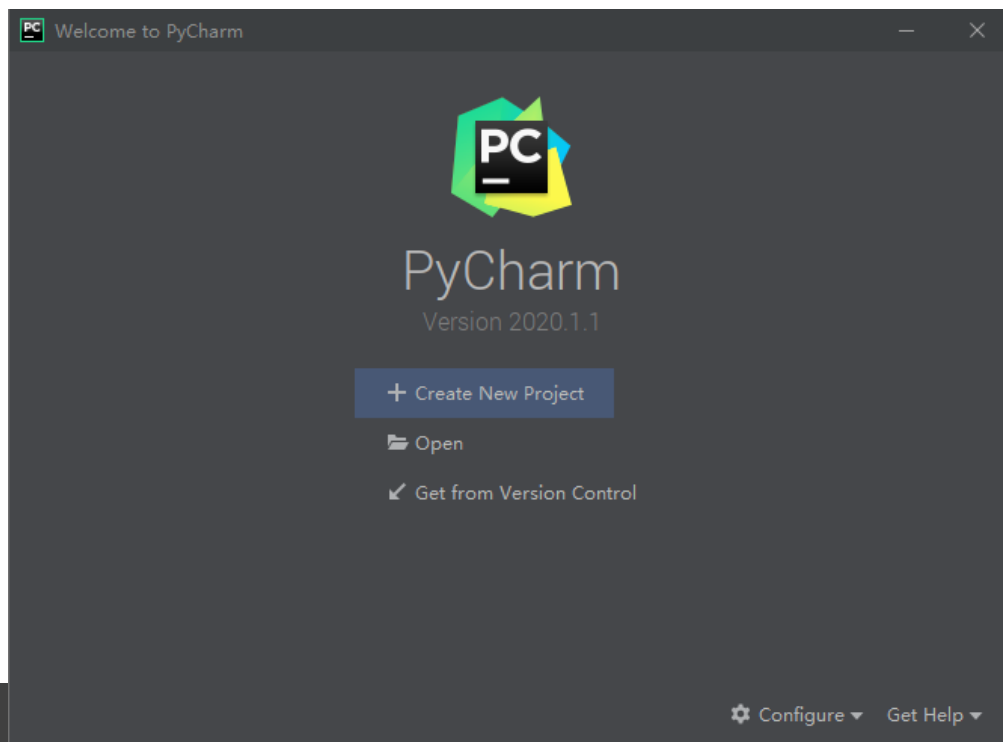
从Windows开始菜单中找到JetBrains文件夹，点击PyCharm Community Edition 2020.1.1菜单项启动PyCharm，首先确认用户协议和决定是否分享软件使用数据。



1.4.2 PyCharm的启动

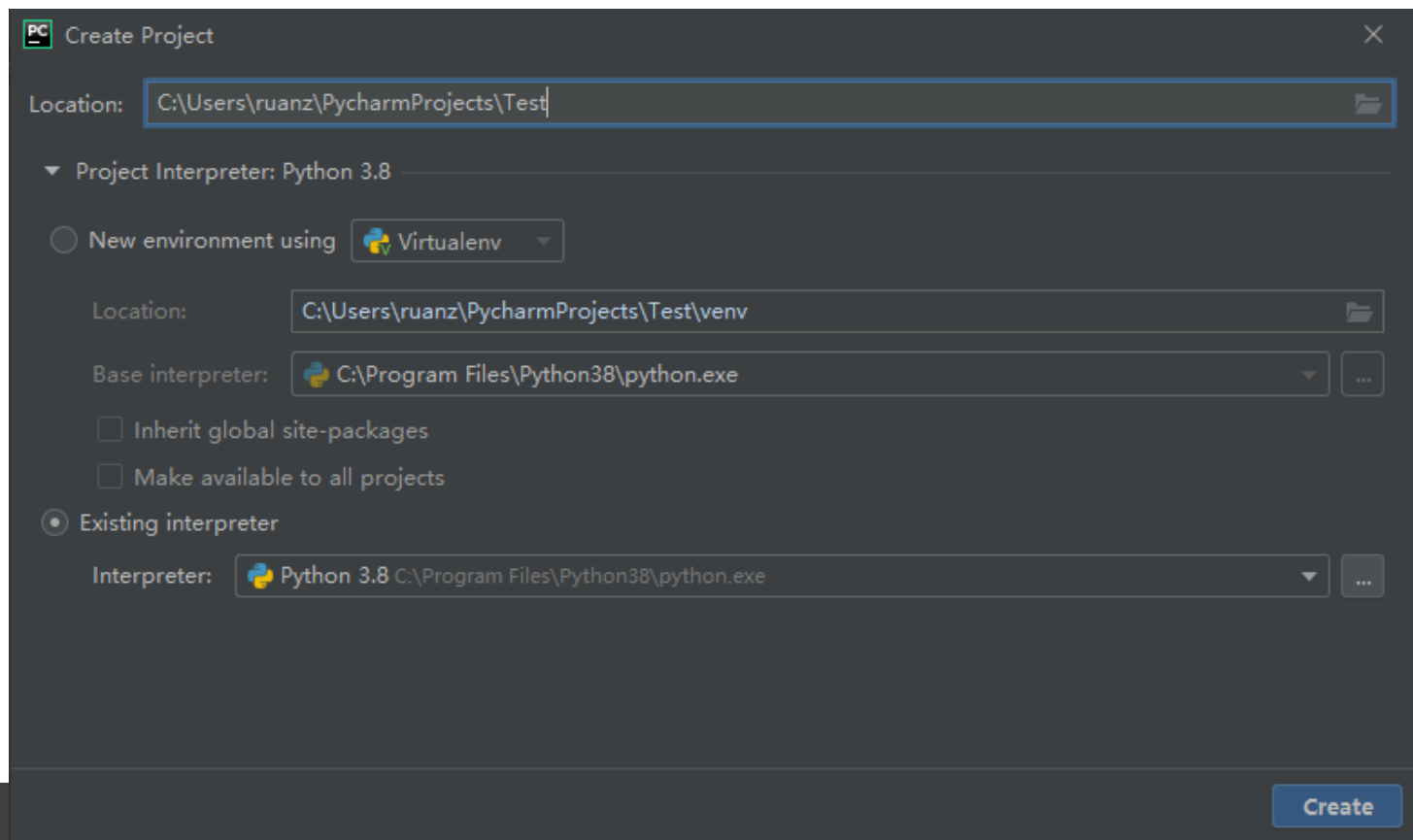


同意用户协议之后，进入到PyCharm启动界面，如左图，完成启动加载过后进入PyCharm欢迎界面，之后便可以创建Python项目，如下图。



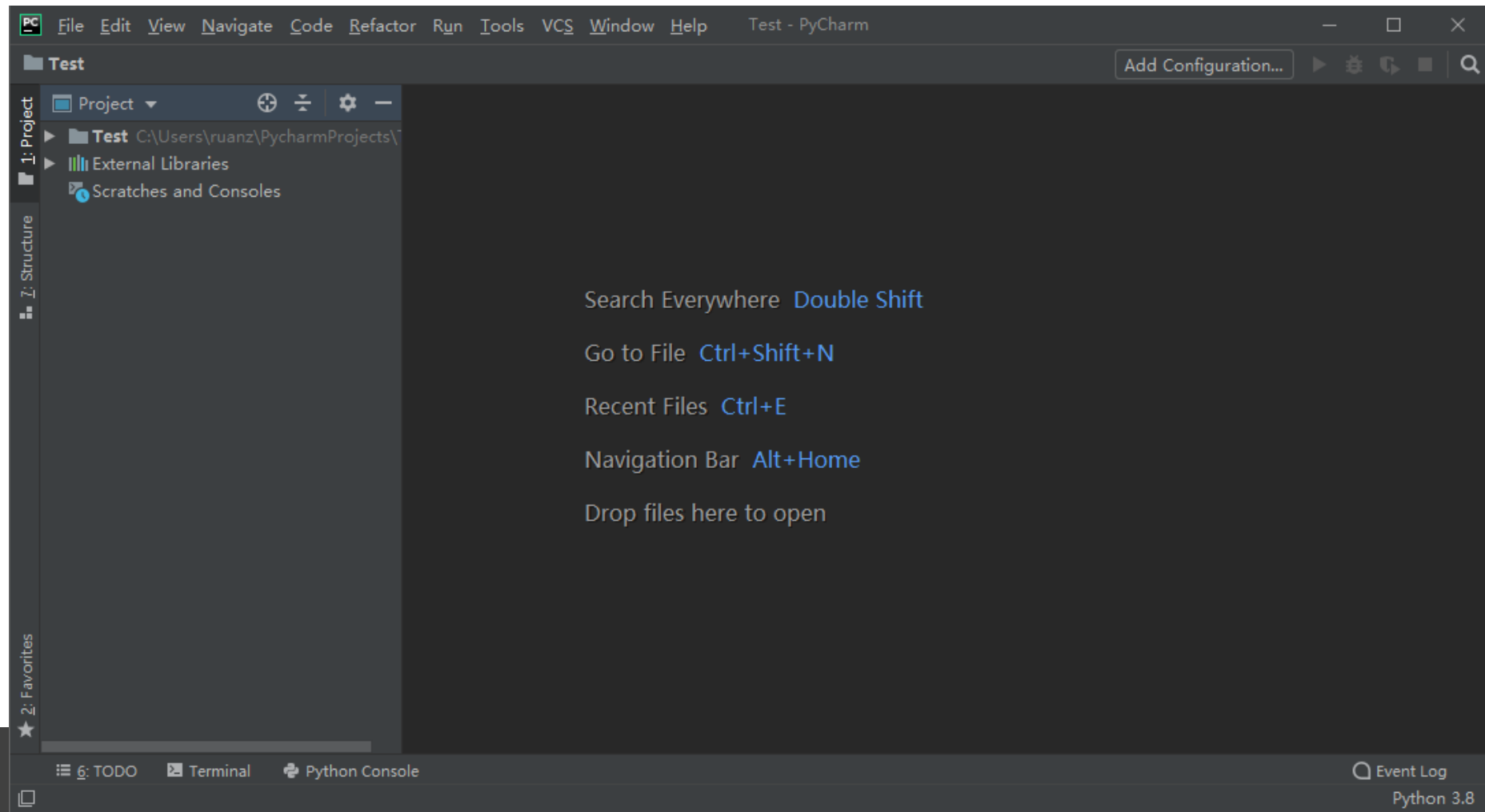
1.4.3 实例：节日贺卡

新建项目，输入项目位置，并指定Python脚本解释器（interpreter）。如果系统中已安装Python或Anaconda，则可以指定它们中的某一个作为解释器；否则需要新建虚拟的Python环境。



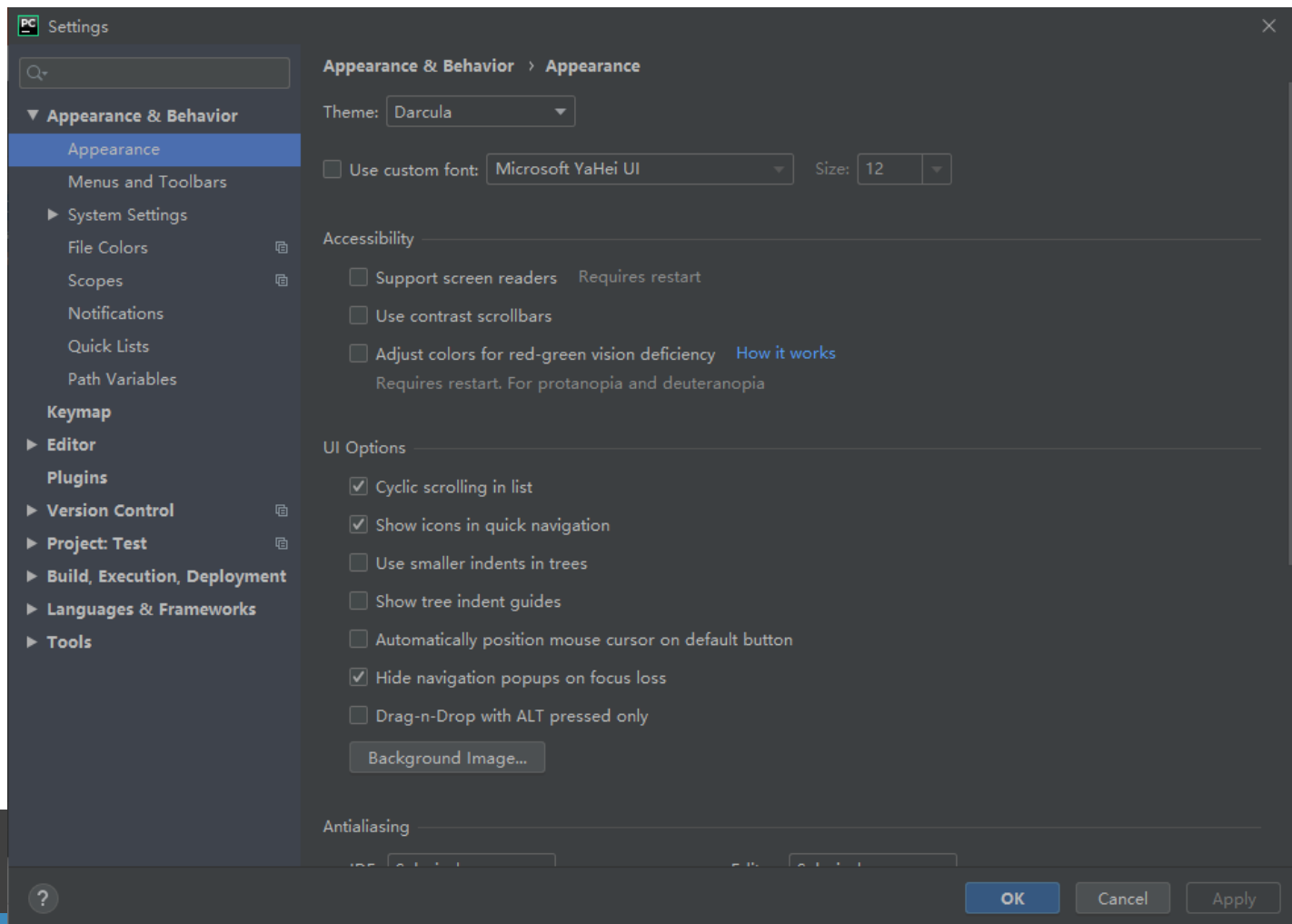
1.4.3 实例：节日贺卡

PyCharm IDE主界面



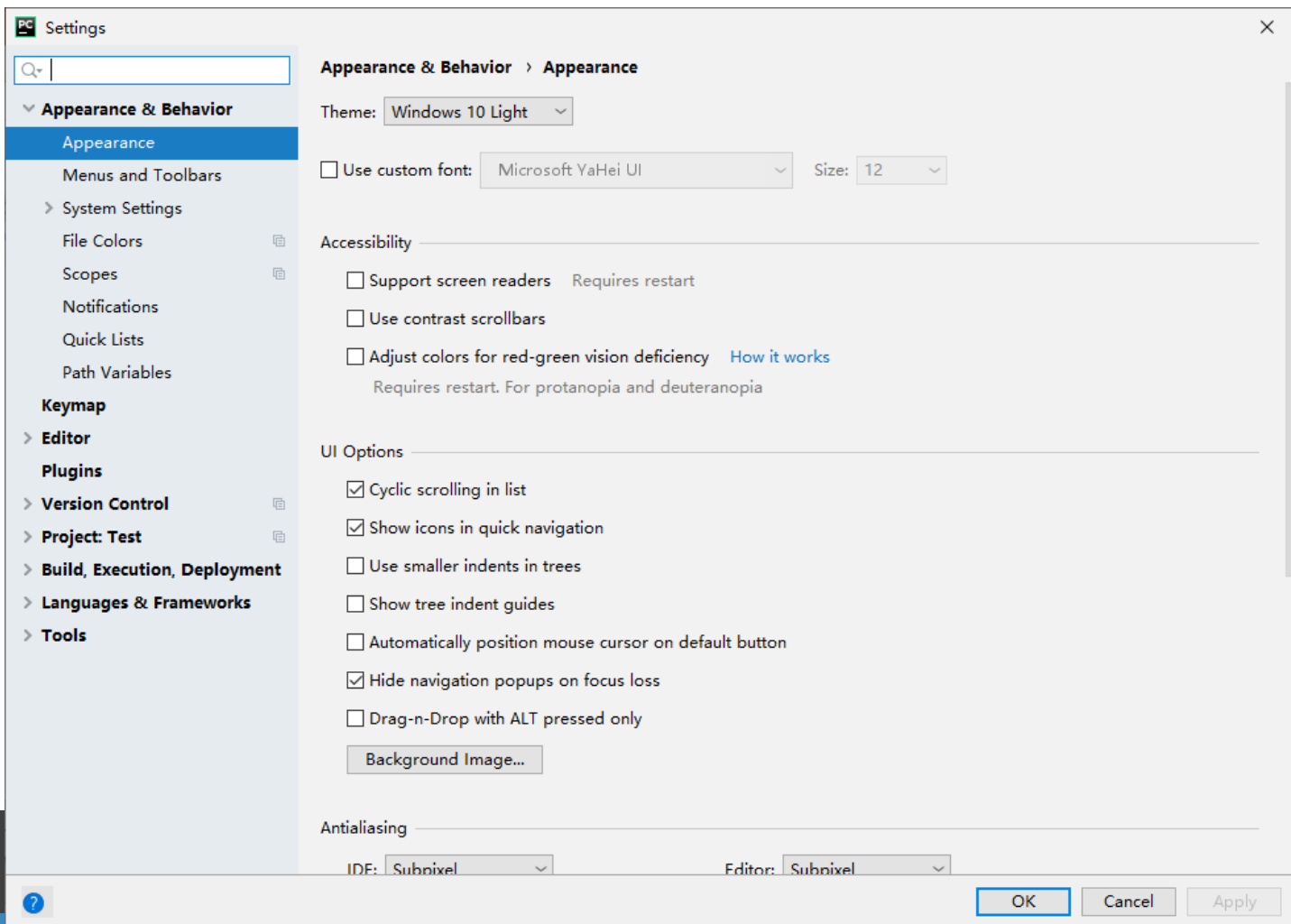
1.4.3 实例：节日贺卡

设置界面主题(Theme): 选择菜单File->Setting, 打开Setting对话框, 找到Appearance可以修改主题



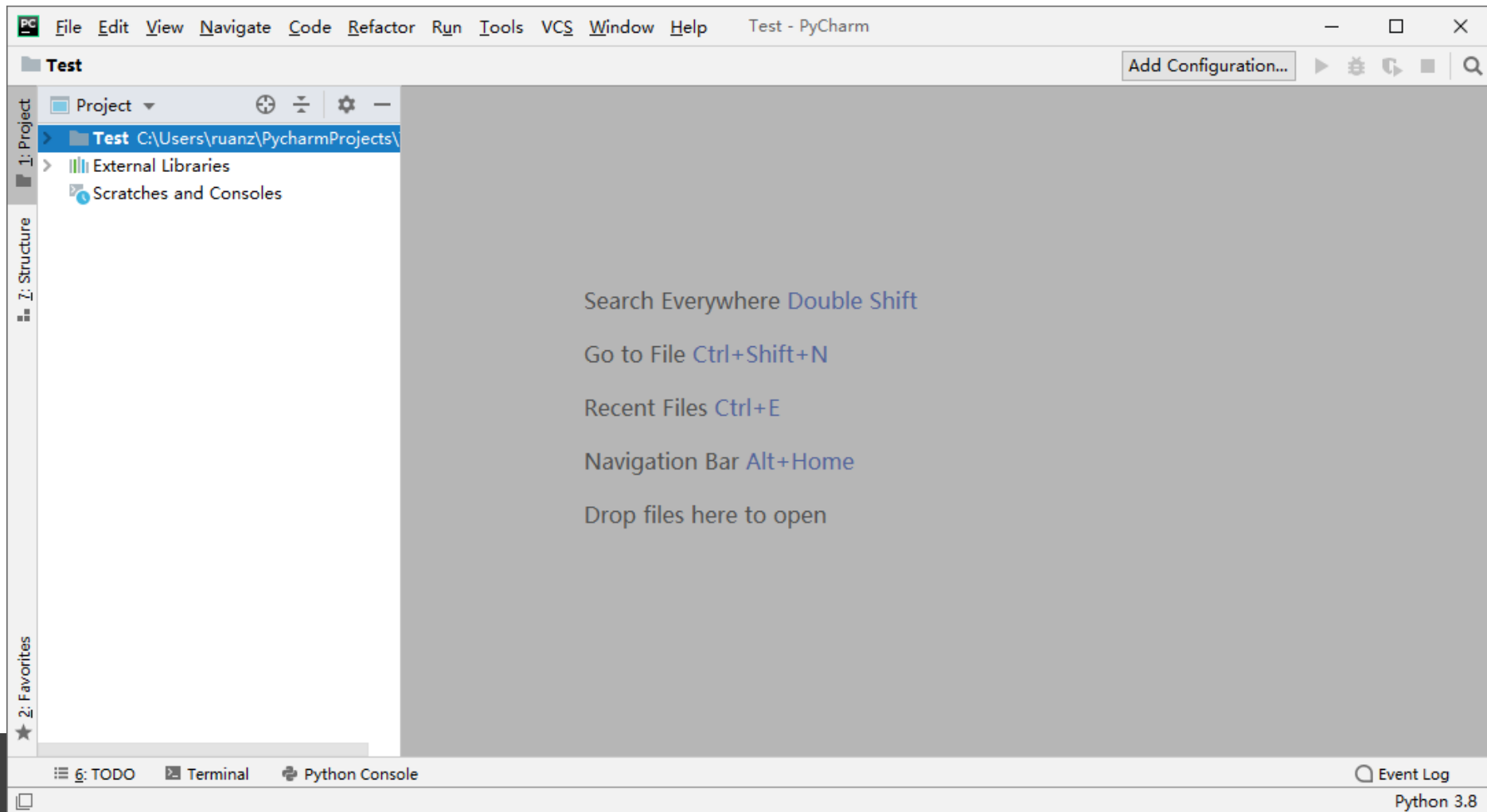
1.4.3 实例：节日贺卡

这里选择“Windows 10 Light”主题(Theme)，然后点击Apply，最后点OK。



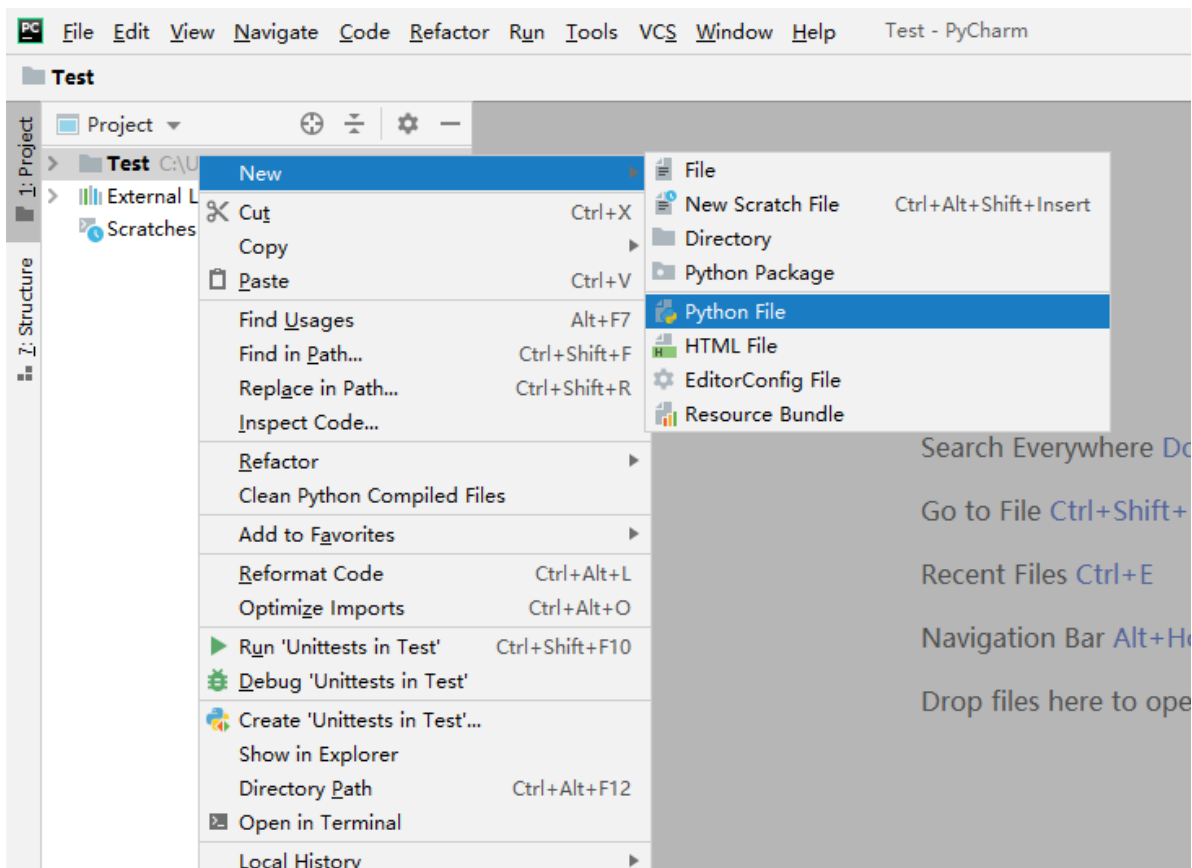
1.4.3 实例：节日贺卡

这里选择“Windows 10 Light”主题(Theme)，然后点击Apply，最后点OK。

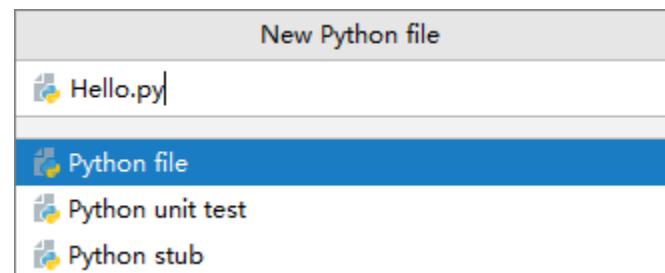


1.4.3 实例：节日贺卡

在项目中新建Python File

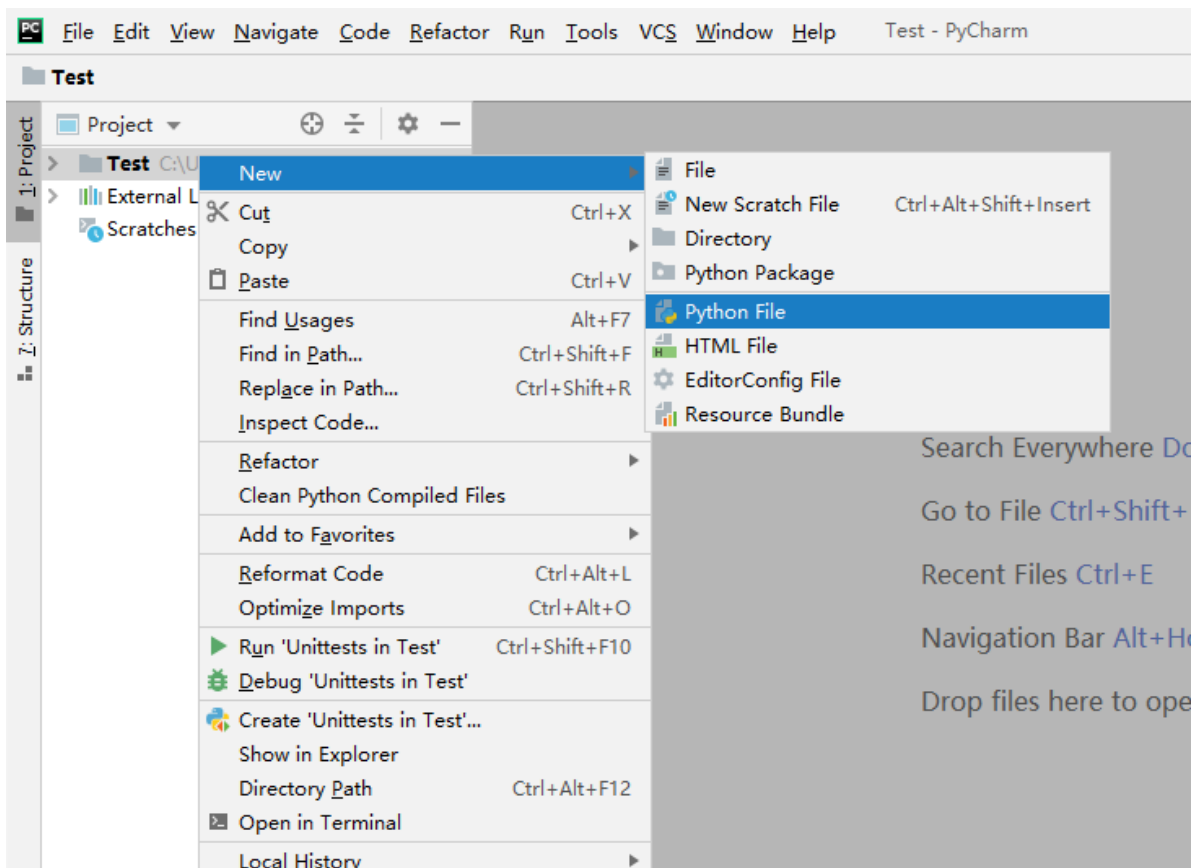


输入文件名并回车

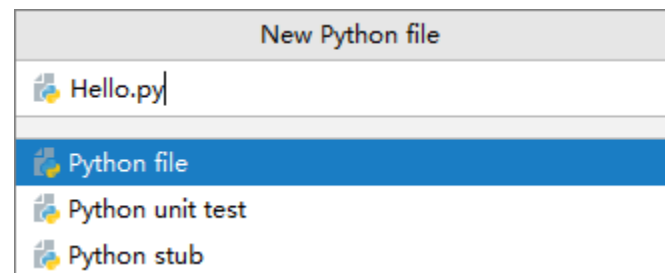


1.4.3 实例：节日贺卡

在项目中新建Python File

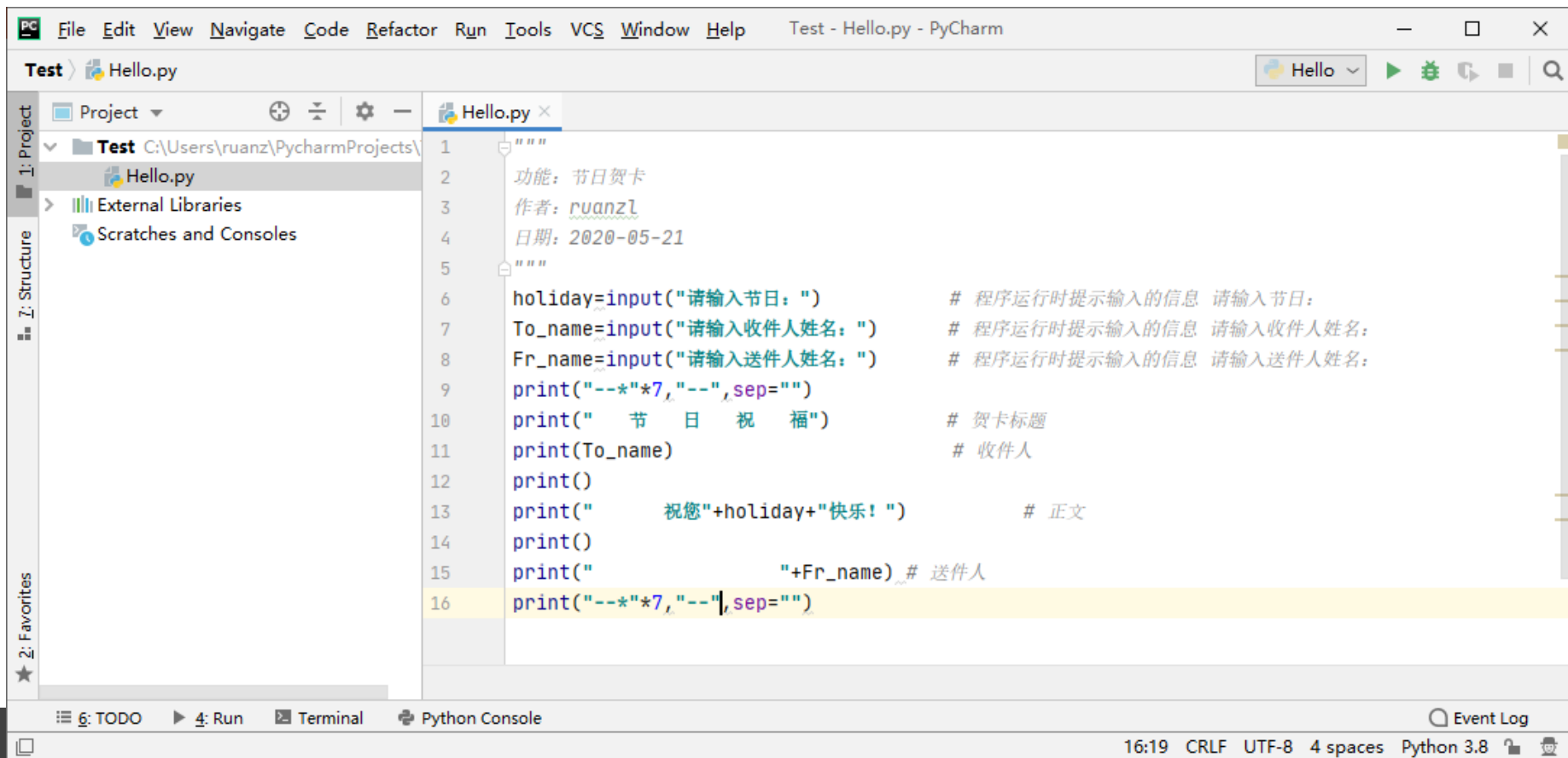


输入文件名并回车



1.4.3 实例：节日贺卡

在PyCharm中编写Python程序。在编写时，要注意添加注释，并注意程序内的缩进。



1.4.3 实例：节日贺卡

点击该窗口右上角的绿色三角形按钮 或选择菜单命令Run->Run 'Hello' , 或按快捷键SHIFT+F10运行程序, 运行结果如下。



```
Run: Hello x
"C:\Program Files\Python38\python.exe" C:/Users/ruanz/PycharmProjects/Test/Hello.py
请输入节日: 国庆节
请输入收件人姓名: 张三
请输入送件人姓名: 李四
-----
      节    日    祝    福
      张  三

      祝您国庆节快乐!

                        李  四
-----
Process finished with exit code 0
```



1.4.4 程序剖析

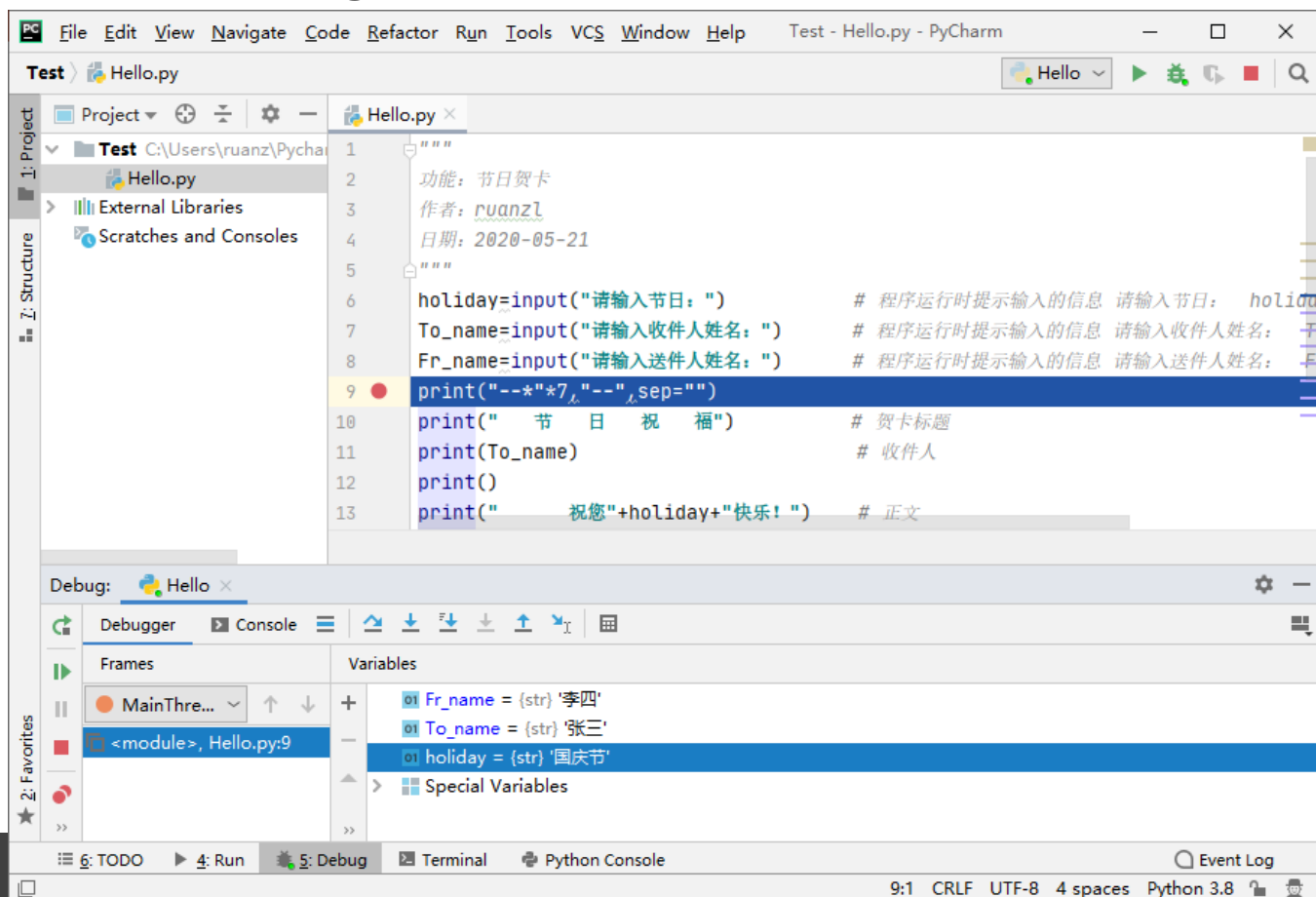
在该程序文件的开头编写了注释，简单说明了我们编写的这个程序要实现的功能、编写者和编写时间等，便于今后对这个程序的后续修改和维护。

然后，定义了三个变量：`holiday`、`To_name`和`Fr_name`，且使用`input()`函数接收键盘输入的字符，为它们赋值。最后，使用`print()`函数输出贺卡内容，`print()`没有参数时输出空行。

在每个代码行后面，我们使用了“`#`”添加注释，是为了便于大家的阅读和理解，不过实际项目中可能不需要在每行都添加注释。

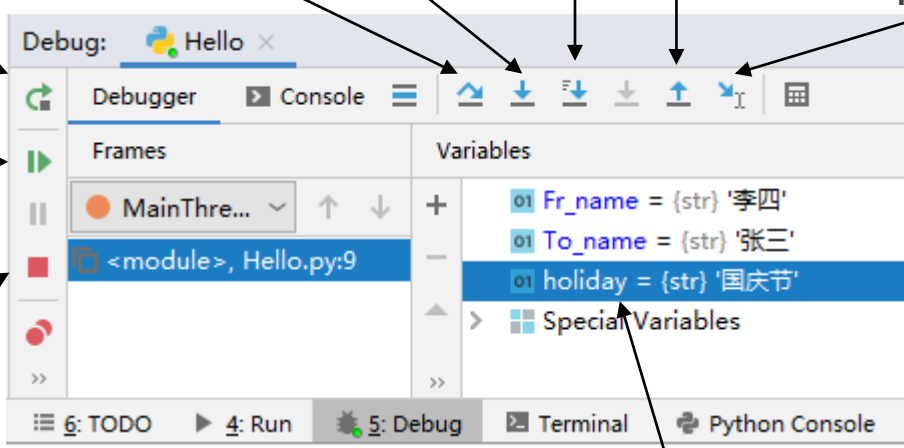
1.4.5 程序调试 (Debug)

行号右侧空白处单击设置中断点，然后点击该窗口右上角的绿色小虫按钮 ，或选择菜单命令Run->Debug 'Hello'，或按快捷键SHIFT+F9启动程序调试



1.4.5 程序调试 (Debug)

点击该窗口右上角的绿色小虫按钮 ，或选择菜单命令Run->Debug 'Hello'，或按快捷键SHIFT+F9启动程序调试



The image shows the PyCharm Debug window with several annotations:

- Step Over**: Points to the first icon in the toolbar.
- Step Into**: Points to the second icon in the toolbar.
- Step Into My Code**: Points to the third icon in the toolbar.
- Step Out**: Points to the fourth icon in the toolbar.
- Step To Cursor**: Points to the fifth icon in the toolbar.
- Rerun 'Hello' (重新开始调试)**: Points to the green bug icon in the top right of the window.
- Resume Program (继续, 直到下一个中断点)**: Points to the green play button icon in the toolbar.
- Stop 'Hello'**: Points to the red square icon in the toolbar.

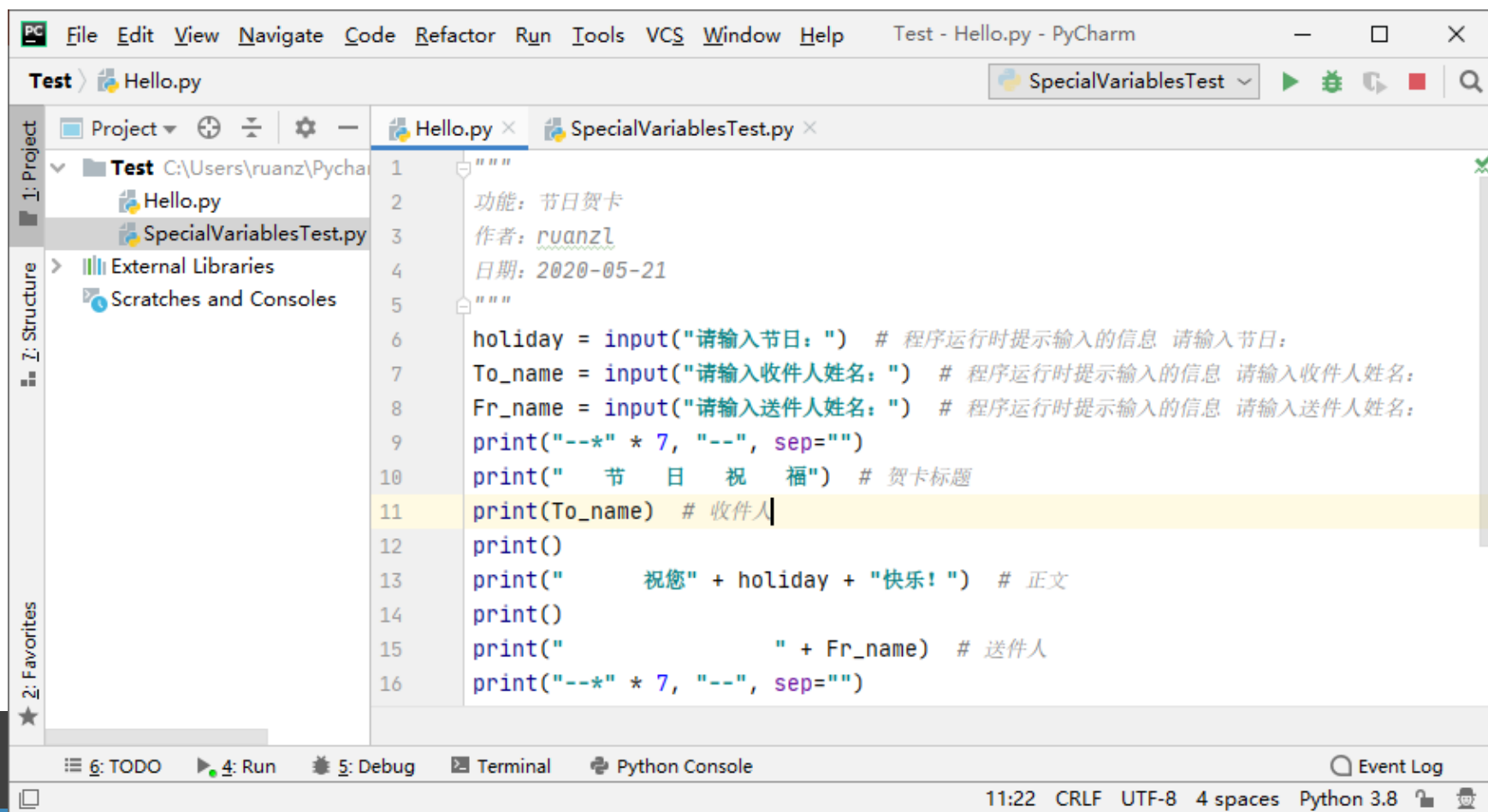
The **Variables** panel on the right shows the following values:

Variable	Value
Fr_name	{str} '李四'
To_name	{str} '张三'
holiday	{str} '国庆节'
Special Variables	

当前变量的值，鼠标移至程序某个变量上，也会显示其值

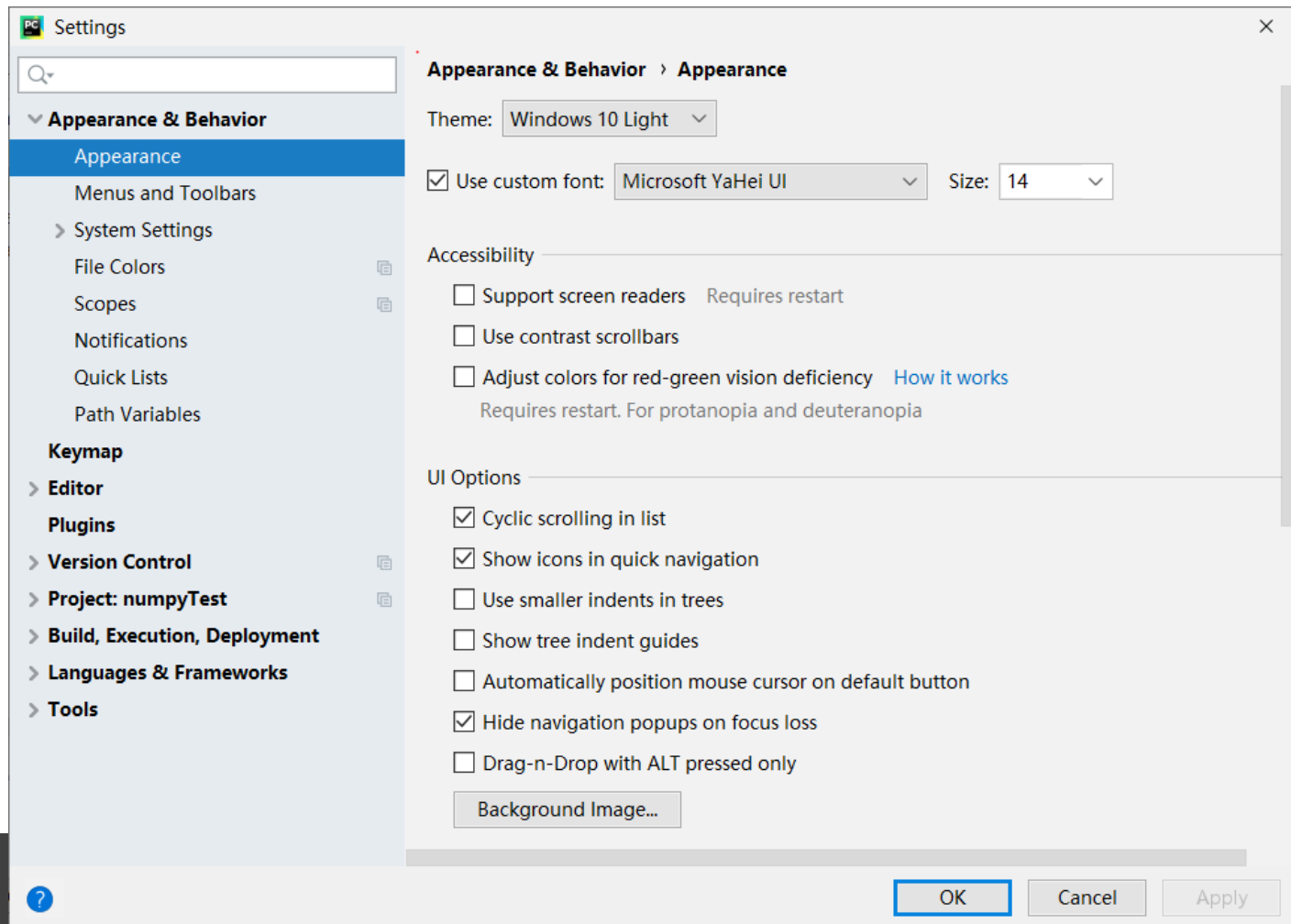
1.4.6 格式化代码

选择需要格式化的代码，然后选择菜单命令Code->Reformat Code,或按快捷键Ctrl + Alt + L，进行代码格式化。或者选择菜单命令Code->Reformat File格式化文件



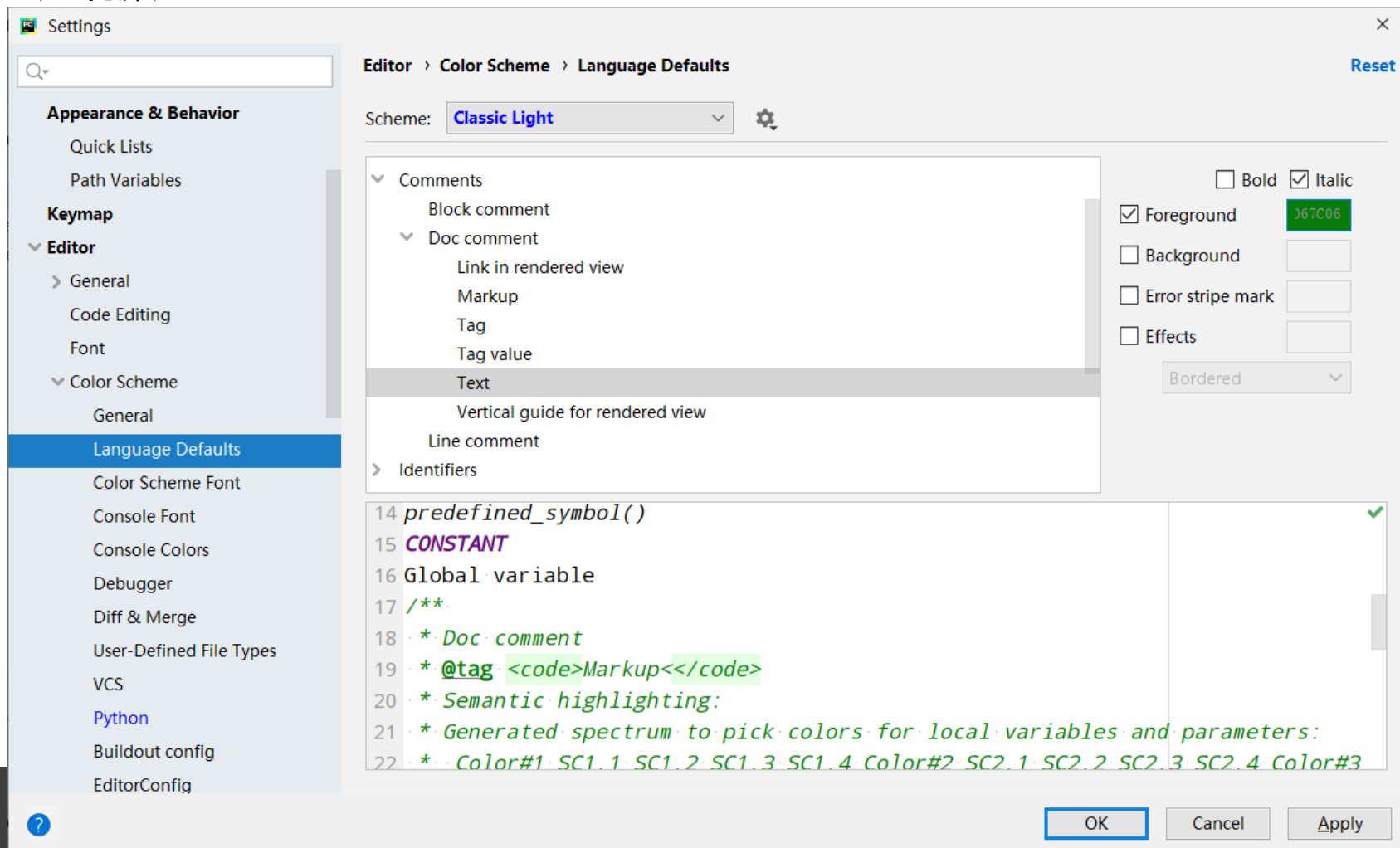
1.4.7 设置代码风格、字体、颜色

IDE外观主题及字体



1.4.7 设置代码风格、字体、颜色

定制颜色

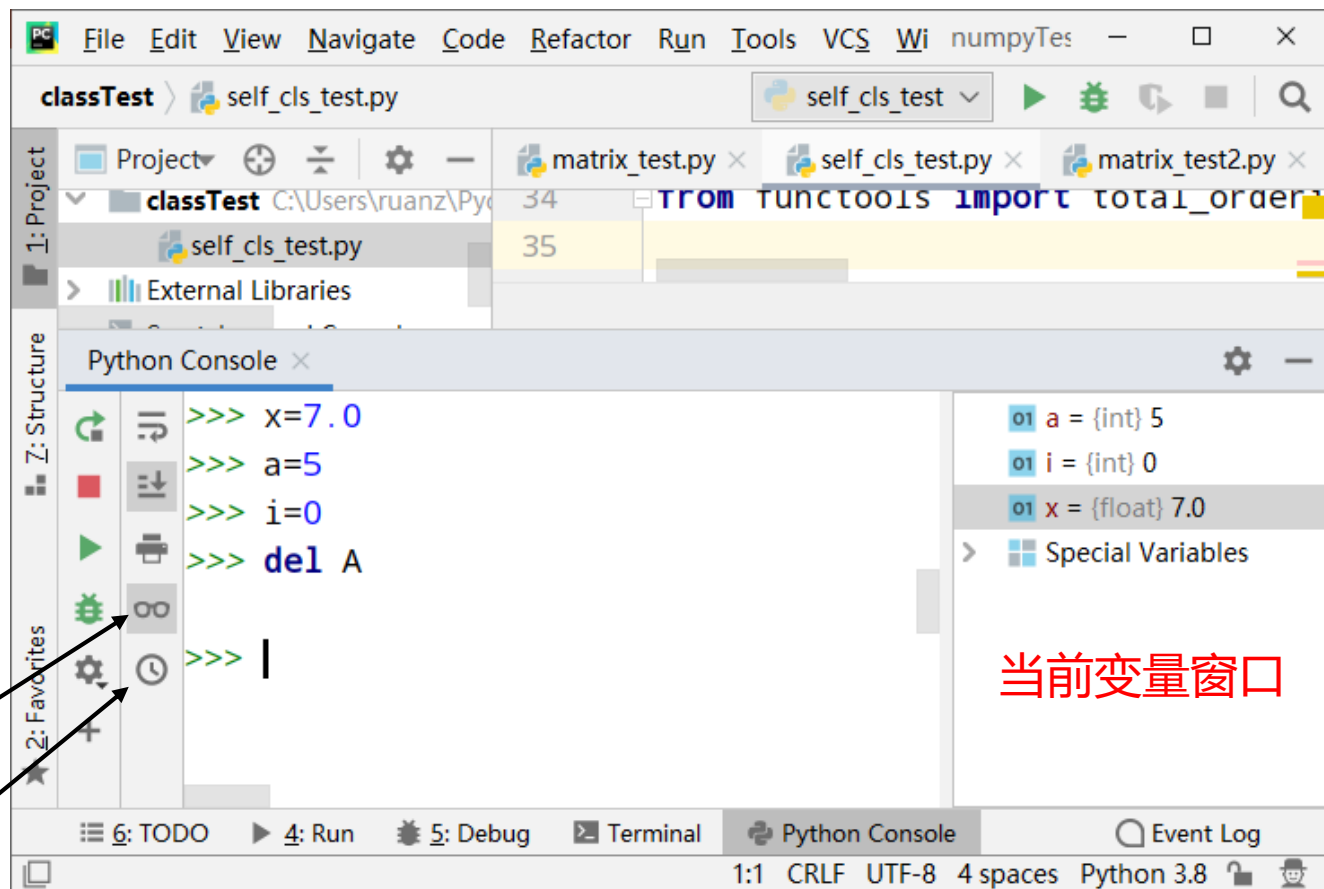


1.4.8 使用Python命令行

- 向上、向下键—取出历史命令
- Tab键—显示智能提示列表
- del 命令—删除变量 (对象)

显示当前变量窗口

显示历史命令窗口



第一章 Python3概述

1.1 Python 简介

1.2 Python环境构建

1.3 第一个程序 Hello World !

1.4 使用 IDE PyCharm

1.5 小结

习题

本章简单介绍了Python 的发展历程和特性，Python版本的选择，Python 3.8.3版本在主流操作系统上的安装方法，Python自带的集成开发和学习环境IDLE的使用方法，以及另一种Python集成开发环境PyCharm的安装和使用。

根据IEEE Spectrum于2017年发布的一份研究报告显示，**Python 超越了C和Java，成为世界上最受欢迎的编程语言**。重要的是，Python这种**简单加愉快**的编程语言，在全球掀起了学习和使用的热潮，越来越多的爱好者和团队加入到使用 Python开发应用程序的行列。可以预见的是，Python语言必将成长得更加强大，更加精致，更加完美。

第一章 Python3概述

1.1 Python 简介

1.2 Python环境构建

1.3 第一个程序 Hello World !

1.4 使用 IDE PyCharm

1.5 小结

习题

习题：

1. 简述Python语言的设计特点。
2. 简述Python2.X和Python3.X的区别。
3. 熟悉Python开发环境的搭建与配置，并用其编写、调试与运行Python脚本程序

感谢聆听

