

python初级教程：入门详解

版本：v1.4.1

Crifan Li

摘要

本文是针对Python的初学者，从无到有的介绍Python语言如何入门，主要包括了：Python的简介，如何下载Python，如何安装Python，如何使用终端、Shell，IDE等各种开发环境进行Python开发，Python中的语法和基本知识、概念和逻辑，以及继续深入学习Python的方法，另外还整理一些值得参考的资料。



本文提供多种格式供：

| | | | | | | | |
|-------------|-----------------------------------|------------------------------------|-----------------------------------|-----------------------------------|-----------------------------------|-----------------------------------|---------------------------------------|
| 在线阅读 | HTML ¹ | HTMLs ² | PDF ³ | CHM ⁴ | TXT ⁵ | RTF ⁶ | WEBHELP ⁷ |
| 下载（7zip压缩包） | HTML ⁸ | HTMLs ⁹ | PDF ¹⁰ | CHM ¹¹ | TXT ¹² | RTF ¹³ | WEBHELP ¹⁴ |

HTML版本的在线地址为：

http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/html/python_beginner_tutorial.html

有任何意见，建议，提交bug等，都欢迎去讨论组发帖讨论：

http://www.crifan.com/bbs/categories/python_beginner_tutorial/

修订历史

| | | |
|--|------------|-----|
| 修订 1.4.1 | 2015-05-26 | crl |
| 1. 把之前教程的地址移过来 | | |
| 2. 合并帖子的内容：【整理】【多图详解】如何在Windows下开发Python：在cmd下运行Python脚本+如何使用Python Shell（command line模式和GUI模式）+如何使用Python IDE | | |
| 3. 为出版而整理：每章都加了摘要，每章都完善了架构。 | | |
| 4. 整理章节的架构；整理第一章的内容 | | |
| 5. 添加Python学习资料 | | |

¹ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/html/python_beginner_tutorial.html

² http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/htmls/index.html

³ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/pdf/python_beginner_tutorial.pdf

⁴ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/chm/python_beginner_tutorial.chm

⁵ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/txt/python_beginner_tutorial.txt

⁶ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/rtf/python_beginner_tutorial.rtf

⁷ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/webhelp/index.html

⁸ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/html/python_beginner_tutorial.html.7z

⁹ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/htmls/index.html.7z

¹⁰ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/pdf/python_beginner_tutorial.pdf.7z

¹¹ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/chm/python_beginner_tutorial.chm.7z

¹² http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/txt/python_beginner_tutorial.txt.7z

¹³ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/rtf/python_beginner_tutorial.rtf.7z

¹⁴ http://www.crifan.com/files/doc/docbook/python_beginner_tutorial/release/webhelp/python_beginner_tutorial.webhelp.7z

python初级教程：入门详解:

Crifan Li

版本：v1.4.1

出版日期 2015-05-26

版权 © 2015 Crifan, <http://crifan.com>

本文章遵从：[署名-非商业性使用 2.5 中国大陆\(CC BY-NC 2.5\)](#)¹⁵

¹⁵ http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#cc_by_nc

目录

| | |
|---|------|
| 前言 | viii |
| 1. 本文目的 | viii |
| 2. 看此文之前你所要掌握的知识 | viii |
| 1. Python简介 | 1 |
| 1.1. 什么是Python | 1 |
| 1.1.1. Python这个单词的本意是蟒蛇 | 1 |
| 1.1.2. Python是一种计算机语言 | 1 |
| 1.1.3. 作为计算机语言的Python的各种叫法的含义 | 1 |
| 1.1.3.1. Python是一种脚本语言 | 2 |
| 1.1.3.2. Python是一种交互性的语言 | 2 |
| 1.1.3.3. Python是一门解释性的语言 | 2 |
| 1.1.3.4. Python是一种面向对象的语言 | 2 |
| 1.1.3.5. Python是一种高级语言 | 2 |
| 1.2. Python能干什么 | 3 |
| 1.3. Python的特点 | 3 |
| 1.3.1. 作为解释性语言的Python的优缺点 | 3 |
| 1.3.1.1. Python的优点 | 3 |
| 1.3.1.2. Python的缺点 | 3 |
| 1.3.2. Python语言自身的特点 | 3 |
| 1.4. Python相关的必备知识 | 4 |
| 1.4.1. Python文件的后缀是py | 4 |
| 1.4.2. Python的缩写和简称 | 4 |
| 1.4.3. Python的官网 | 4 |
| 1.4.4. Python的logo图案 | 4 |
| 2. 选择并下载合适的版本的Python | 6 |
| 2.1. 为何要注意选择合适版本的Python | 6 |
| 2.2. 如何选择正确版本的Python | 6 |
| 2.2.1. 明确当前所处平台版本的信息 | 6 |
| 2.2.2. 选择可用和速度快的下载源 | 6 |
| 2.2.3. 选择合适的版本的Python | 7 |
| 3. 如何安装Python | 8 |
| 3.1. 如何在Windows系统中安装Python | 8 |
| 3.1.1. 在Win7中安装Python | 8 |
| 3.2. 如何在Linux系统中安装Python | 8 |
| 3.2.1. 在Ubuntu中安装Python | 8 |
| 3.3. 如何在Mac中安装Python | 9 |
| 4. 选择合适的Python开发环境 | 10 |
| 4.1. 如何在Windows环境下开发Python | 10 |
| 4.1.1. Python的最原始的开发方式是什么样的 | 12 |
| 4.1.1.1. 找个文本编辑器，新建个.py文件，写上Python代码 | 12 |
| 4.1.1.2. 打开Windows的cmd，并且切换到对应的python脚本所在目录 | 17 |
| 4.1.1.2.1. 方法1：手动打开cmd，并cd到对应路径 | 17 |
| 4.1.1.2.2. 方法2：通过Notepad++的Open current dir cmd | 17 |
| 4.1.1.3. 在cmd中去运行你的Python脚本（.py文件） | 18 |
| 4.1.2. 利用Python的shell进行交互式开发又是什么样的 | 20 |
| 4.1.2.1. 命令行版本的Python Shell – Python (command line) | 20 |
| 4.1.2.2. 带图形界面的Python Shell – IDLE (Python GUI) | 22 |
| 4.1.2.3. 关于（command line或GUI版本的）Python Shell的用途 | 28 |
| 4.1.3. 利用第三方Python的IDE进行Python开发又是怎么回事 | 28 |
| 4.1.3.1. 为何会有Python的IDE | 29 |
| 4.1.3.2. 目前常见的一些Python的IDE | 29 |
| 4.1.3.3. Python的IDE和Python代码编辑器，Windows的cmd，等的关系 | 30 |
| 4.1.3.4. 使用IDE时所遇到的一些常见的问题 | 31 |
| 4.1.3.4.1. IDE只能够打开了文件，并不代表就已经在shell中运行了该文件 | 31 |

| | |
|---|----|
| 4.1.3.4.2. 需要注意，确保有可以运行的Python起始部分的代码 | 33 |
| 4.1.4. 总结：到底使用哪种环境去开发Python | 34 |
| 4.1.4.1. 对初学者的建议：如何选用Python的开发环境 | 34 |
| 4.1.5. 如何在Windows环境下使用Python脚本 | 34 |
| 4.1.5.1. 如何在Windows下的cmd中运行BlogsToWordpress.py | 36 |
| 4.2. 如何在Linux环境下开发Python | 38 |
| 4.3. 如何在Mac环境下开发Python | 38 |
| 5. Python的基本语法和基础知识 | 39 |
| 5.1. 一张图片入门Python | 39 |
| 5.2. Python中的2.x版本和3.x版本 | 41 |
| 5.3. Python文件编码声明 | 42 |
| 5.4. Python中的缩进 | 42 |
| 5.5. Python中基本变量的声明和定义 | 42 |
| 5.5.1. Python中变量的作用域 | 42 |
| 5.5.2. Python中变量与C语言中的变量对比 | 42 |
| 5.6. Python中的分支结构 | 43 |
| 5.7. Python中的函数 | 43 |
| 5.8. Python中的面向对象编程 | 43 |
| 6. 继续学习Python的思路和方法 | 44 |
| 6.1. 如何继续深入学习Python | 44 |
| 6.2. 如何利用Python相关资源 | 45 |
| 6.2.1. 如何利用Python自带的手册 | 45 |
| 6.2.2. 如何利用一些在线的Python资源 | 45 |
| 7. Python常见问题及解答 | 46 |
| 常见问题 | 46 |
| 7.1. 在window的cmd中运行python结果却调用了文本编辑器去打开了，而不是去调用Python解析器去运行python文件 | 46 |
| 8. Python相关资源 | 47 |
| 参考书目 | 48 |

插图清单

| | |
|---|----|
| 4.1. 在Windows下的cmd下面运行Python脚本的样子 | 35 |
| 4.2. 开始菜单中找到的Python (Command Line) | 36 |
| 4.3. Python (Command Line)的界面 | 36 |
| 4.4. 动画演示如何在Windows的cmd中运行Python脚本BlogsToWordpress.py | 37 |
| 4.5. 在Mac下的Terminal中运行Python脚本：BlogsToWordpress | 38 |
| 5.1. Quick Python Script Explanation | 40 |
| 5.2. 一张图入门Python中文版 | 41 |
| 7.1. 安装Python时选择Register Extensions | 46 |

范例清单

| | |
|-----------------------------------|----|
| 4.1. 举例：用Python的IDLE去做URL解码 | 28 |
|-----------------------------------|----|

公式清单

| | |
|--------------------------|----|
| 4.1. 什么是IDE | 29 |
| 4.2. 什么是Python的IDE | 29 |

前言

1. 本文目的

本文目的在于，让原先对于Python不熟悉的，甚至没什么概念的人。

看完本系列教程后，从Python的小白，变成，对于Python可以算是入门了。

2. 看此文之前你所要掌握的知识

最好有其他计算机语言的基础

比如C语言，Java语言等。

当然，如果没有，也是可以看此文的。

我会在必要的时候，进行相应的提示的。

第 1 章 Python简介

本章主要讲解在开始学习Python之前，所需要了解到的Python的一些基本概念，包括Python是什么，Python的特点和常见的用途，以及Python的两大版本之间的主要区别。

在开发Python之前，需要先搞懂Python是啥。

下面就是介绍一下，概念性的东西。

[【整理】Python语言简介](#)¹

1.1. 什么是Python

1.1.1. Python这个单词的本意是蟒蛇

Python，首先，作为一个英文单词，其本意是：

巨蟒，蟒蛇

的意思。

1.1.2. Python是一种计算机语言

对于，Python，这个词来说，在计算机语言领域内，此处，我们指的是，一种计算机语言，叫做Python



Python语言的名字的由来

之所以，我们把Python，蟒蛇，作为此计算机语言的名字，是有其历史典故的：

即，Python语言名称的由来的历史了：

Python语言的创始人，吉多·范罗苏姆（Guido van Rossum）

在1989年圣诞节期间，在阿姆斯特丹，为了打发圣诞节的无趣，决心开发一个新的脚本解释程序，

而在给自己新创造的计算机语言起名字的时候，

由于其是，自于七十年代风靡全球的英国六人喜剧团体，巨蟒剧团（Monty Python）的忠实粉丝，

所以，就把此计算机语言的名字叫做：Python

这就是，Python，一种计算机语言，的名字的由来，被叫成了一种看似很凶猛的动物，蟒蛇，的名字。

当然，Python语言本身，并不“凶猛”

倒是，说起Python语言的功能强大，倒是可以真的强大到，称其为，“凶猛”，都不为过的。

1.1.3. 作为计算机语言的Python的各种叫法的含义

作为一种计算机语言的Python，自然也会有，根据计算机语言不同的层面，去将其分类，所以你常常会听到如下说法：

¹ http://www.crfan.com/simple_intro_what_is_python_and_how_to_run_python_script

比如：

http://zh.wikipedia.org/wiki/Python_%28%E6%B6%88%E6%AD%A7%E4%B9%89%29

中的：

Python，一种直译式、交互式、面向对象的计算机编程语言，由Guido van Rossum创建并以Monty Python飞行马戏团的名字命名。

1.1.3.1. Python是一种脚本语言

脚本，对应的英文是：script

普通人，看到script这个英文单词，或许想到的更多的是：电影的剧本，就是一段段的脚本，所组成的电影剧本的脚本，决定了电影中的人和物，都做哪些事情，怎么做

而计算机中的脚本，决定了：

计算机中的操作系统和各种软件工具，要做哪些事情，以及具体怎么做

其实，脚本，这个词，笔者的理解，还有另外一种感觉：随性

因为，现实中，写电影脚本，那直接拿张纸和笔，就可以写了，写完了，就可以拿去用，拿去拍电影了

而计算机中的脚本，其实就是普通的文本，就是写到文本文件中的代码而已，其也是有随性这个特点：写完了，直接就可以（在某种具有解释功能的环境中）运行了

比如：Linux中的shell脚本，就是直接弄个文本文件，写上shell脚本代码，然后保存文件，然后就可以，在Linux的Shell这个带有shell的解释功能的shell环境中，运行了

与此相对的，不那么随性的是：其他的，非脚本语言，常叫做编译性语言，比如C语言，往往都是需要额外加上编译这个步骤，才能执行的。

1.1.3.2. Python是一种交互性的语言

1.1.3.3. Python是一门解释性的语言

解释性，也被叫做：直译式



什么是解释性语言

简单说就是，无需编译源码为可执行文件，直接使用源码，就可以运行的语言。

此处即，对于Python的源代码，文本文件，直接就可以去执行了，不需要像C，C#等语言，还需要编译一下，才可以运行。

更详细的解释，参见：TODO：解释性语言的详细解释。

1.1.3.4. Python是一种面向对象的语言

1.1.3.5. Python是一种高级语言

<http://www.baike.com/wiki/Python>

1.2. Python能干什么

这里：

[Python - 维基百科，自由的百科全书](#)²

中就有很多举例：使用Python编写的著名应用

1.3. Python的特点

1.3.1. 作为解释性语言的Python的优缺点

而Python语言，本身就是一种解释性语言，

所以，作为解释性语言的Python，本身就有解释性语言天然就有的一些优点和缺点：

1.3.1.1. Python的优点

- 不需要编译，即可运行

1.3.1.2. Python的缺点

- 性能相对不是特别强
所谓的，性能不是特别强，是先对来说的

在某些，或者说，极少的，领域，对于性能要求非常高，则此时就不太适合使用Python去实现了

现在已有的解决方法是：使用其他，相对Python性能更好的语言，实现和性能相关的那部分的功能，

然后再整合集成到Python内。

1.3.2. Python语言自身的特点

除了作为解释性语言的Python自带的优缺点之外，作为Python语言本身，的确有其自己的特点：

一般来说，讨论某种东西的特点，都是将其和同类事物去比较的；

相应的，此处，讨论Python的特点的时候，把Python和其他计算机语言相比的，其特点，可以描述如下：

Python相对最大的一个特点，算是：

代码缩进决定了代码的逻辑关系，而不仅仅是为了好看

其他更多的特点，可以总结如下：

- 有很多特性，使得语言本身很方便编程者使用
比如对于字符串可以直接通过数组下表去获得对应子字符串

² <http://zh.wikipedia.org/wiki/Python>

这类特性，是很多其他语言，本身所不具有的。

- Python是一种被广泛采用的胶水语言
粘合能力，相对其他语言，很强
- 功能丰富的自带的库以及种类和数量繁多且强大的第三方库
除了本身Python语言本身自带的标准库之外

由于Python语言本身很好用，促使很多其他人，前前后后，写了N多个库

积累至今，就成了：Python拥有数量巨大的，各种各样的，功能强大的，第三方的库函数。

此特点，对于Python语言的使用者来说，有着显而易见的好处：

当你去实现很多各种功能的时候，往往不需要重复造轮子，

往往是可以找到，现成的，已有的，内置Python库，或第三方的Python的库，去帮你实现你要的功能

换句话说：你去实现一个复杂的功能的时候，往往变成了，找到合适的库，并使用，即可。

与此相对的，用其他语言去实现同样的功能的时候，由于缺少好用的库，而使得你需要重头到尾，全新的实现对应的功能所需的代码。

由此对比出，Python的好用和强大。

1.4. Python相关的必备知识

对于Python语言的学习，即使没有去深入学习Python的细节知识

对于Python相关的一些必备知识，常识性的知识，也是需要先去了解清楚的。

下面就是对于Python的相关的常识性的东西，先做个解释：

1.4.1. Python文件的后缀是py

计算机的世界中，多数东西的类型，都是通过文件名的后缀来区分的。

Python的文件的后缀是py

1.4.2. Python的缩写和简称

正是由于Python的文件名后缀是py

所以，很多人，也常常用py来指代Python

比如后面你会提到的，有些人把Python 2简写为py2

1.4.3. Python的官网

<http://www.python.org/>

1.4.4. Python的logo图案

第 2 章 选择并下载合适的版本的Python

本章主要讲解，在开始Python学习之前，如何根据自己的当前的操作系统的版本，去选择和下载合适的版本的Python。

开发Python之前，肯定是要先搞清楚，如何下载对应的Python。

本来下载一个东西是很简单的事情，但是由于Python有很多版本方面的事情，所以需要搞清楚很多基本逻辑和概念，才知道自己需要下载哪个版本的。

待整理：

[【教程】如何下载最新版的，各种版本的，包括Python 2.x和Python 3.x的Python](#)¹

2.1. 为何要注意选择合适版本的Python

因为，如果选择错误的，不合适的版本的Python，

则很可能导致后续学习Python和开发Python时，会遇到问题和错误。

比如，一些相对常见，由于版本选择错误而出现的问题有：

- 由于选择了Python 3.x版本，但是参考别人的Python 2.x的代码，结果会出现语法错误
其中最最常见的，有两个经典的例子：
 - printf的语法不同而导致出错
 - TODO：好像有个string还是其他什么的，版本2.x和版本3.x是不同的

当然，出处之外，Python 2.x和Python 3.x还有其他更多语法上面的不同，再次就不一一细说了。

- x64的系统中安装了x86的库，导致后续使用时无法使用
TODO：找到对应的x64库的错误的例子

2.2. 如何选择正确版本的Python

2.2.1. 明确当前所处平台版本的信息

搞清楚当前操作系统的具体信息

尤其是什么系统，什么版本

以便于后续选择与当前系统所匹配的，最合适的版本

2.2.2. 选择可用和速度快的下载源

Python官网中的下载页面：

<http://www.python.org/ftp/python/>

<http://python.org/getit/releases/>

¹ http://www.crfan.com/tutorial_how_to_download_latest_misc_2_x_3_x_version_python

但是Python官网，不是太稳定：对于国内开发者来说，时不时会出现无法访问的情况。

后来找到一个Python官网的国内的镜像：

<http://mirrors.sohu.com/python/>

速度相对来说，蛮快的。

如果发现Python官网无法访问和下载的话，可以去上述镜像去下载。

2.2.3. 选择合适的版本的Python

接下来，就是去选择，合适自己的，与自己当前的系统所匹配的版本的Python，然后再去下载。

什么叫合适自己的版本的Python呢？

那就是：

明确了之前介绍的，Python有2.x和3.x的区别后，选择对应的2.x或3.x后

再根据当前自己系统，下载对应的版本的Python

第 3 章 如何安装Python

本章主要讲解，在下载了合适的版本的Python后，图文并茂的介绍如何去安装Python。

在下载完毕Python后，就是去安装Python了。

3.1. 如何在Windows系统中安装Python

Windows系列操作系统，有很多不同版本，

目前主流的，最常用的有Windows XP, Windows 7, Windows 8

3.1.1. 在Win7中安装Python

下面以Windows 7为例，来解释如何在Windows中安装Python

待整理：

[【教程】如何在Windows系统中安装Python](#)¹

其中最重要的一点是：最好安装的路径中，不要包含中文

是为了避免以后可能出现的，在Python开发期间，由于中文路径，而导致的一些问题

3.2. 如何在Linux系统中安装Python

Linux系统是个统称

其有很多所谓的发行版，

其中目前相对比较流行一些有：（较早的）RedHat（红帽子），OpenSUSE，以及后来出现的Ubuntu，等等

3.2.1. 在Ubuntu中安装Python

好像是：由于Linux类的系统中，包括Ubuntu，很多内置的工具和服务，都是Python代码写的。

所以，为了保证系统的正常运行，保证不影响当前自己的Linux系统，

最好不要卸载旧版本的Python，

而是在Linux类系统中自带的，已有的Python基础上，安装另外一个，相对最新的Python

当然，其实如何只是为了学习Python，尤其是对于Python初学者来说，其实不必再安装另外一个版本的系统

此处，只是为了，对于以后觉得有必要的时候，比如需要更新版本的Python，利用其中更好的功能等，而去安装另外一个更新版本的Python

下面就来解释，如何在Linux类系统中，安装另外一个，更新版本的Python

¹ http://www.crfan.com/tutorial_how_to_install_python_on_windows

3.3. 如何在Mac中安装Python

由于Mac系统的基本架构，和Linux系统，比较类似

所以，关于如何在Mac上安装Python

其实是和上面已经介绍的，在Linux中安装Python，其原理和步骤，都是类似的

第 4 章 选择合适的Python开发环境

本章主要讲解，在安装了Python之后，针对开发环境的概念，进行深入浅出的介绍，

最原始的开发环境以及常见的IDE开发环境的关系如何，以便使得读者真正了解到各种开发环境的内在关系和优缺点，

才能真正帮助读者选择合适自己的开发环境，提高自己的开发效率。



相关旧帖

- [【整理】Python语言简介](#) ¹
- [【整理】各种Python的IDE\(集成开发环境\)的总结和对比](#) ²
- [【记录】使用Python的IDE：PyScripter](#) ³
- [【已解决】PyScripter启动出错：Python could not be properly initialized. We must quit.](#) ⁴
- [【记录】使用Python的IDE：Ulipad](#) ⁵
- [【已解决】安装Ulipad后，选择启动Ulipad，结果无法启动](#) ⁶
- [【记录】使用Python的IDE：Eclipse+PyDev](#) ⁷
- [【教程】在Eclipse中安装PyDev](#) ⁸
- [【教程】在Eclipse中配置刚安装好的PyDev插件](#) ⁹
- [【教程】在Eclipse中使用PyDev进行Python开发](#) ¹⁰
- [【已解决】把Eclipse中的PyDev中的Python代码中的很难看的中文换个好看点的字体](#) ¹¹
- [【已解决】Eclipse+PyDev无法调试Python：Unexpected IO Exception in Pydev debugger](#) ¹²
- [【记录】折腾IDE工具：Aptana Studio 3](#) ¹³
- [【已解决】Aptana Studio 3中通过Auto Config配置PyDev中Python出错：java.io.IOException: Cannot run program "python": CreateProcess error=2, The system cannot find the file specified](#) ¹⁴

4.1. 如何在Windows环境下开发Python

此部分内容的目的：

¹ http://www.crifan.com/simple_intro_what_is_python_and_how_to_run_python_script

² http://www.crifan.com/summary_common_python_ide_pyscripter_ulipad_eclipse_pydev_eric

³ http://www.crifan.com/try_with_python_ide_pyscripter

⁴ http://www.crifan.com/pyscripter_start_error_python_could_not_be_properly_initialized_we_must_quit

⁵ http://www.crifan.com/try_with_python_ide_ulipad

⁶ http://www.crifan.com/ulipad_after_install_finish_not_launch

⁷ http://www.crifan.com/try_with_python_ide_eclipse_pydev

⁸ http://www.crifan.com/eclipse_install_plugin_pydev

⁹ http://www.crifan.com/eclipse_configure_newly_installed_plugin_pydev

¹⁰ http://www.crifan.com/eclipse_use_pydev_develop_python

¹¹ http://www.crifan.com/eclipse_pydev_change_ugly_zhcn_char_to_another_font

¹² http://www.crifan.com/eclipse_pydev_python_unexpected_io_exception_in_pydev_debugger

¹³ http://www.crifan.com/try_dev_ide_apтана_studio_3

¹⁴ http://www.crifan.com/apatana_studio_3_auto_config_pydev_error_java_io_ioexception_cannot_run_program_python_createprocess_error_2

希望对于，如何在Windows下，写Python代码，进行Python开发，运行Python脚本的人，看了此部分内容后，懂得了：

- 什么是cmd下面去运行Python脚本
- 什么是Python的交互式的shell
- 什么是Python的IDE

看此部分内容之前，需要具有以下前提：

- Python语言的基本知识
包括知道其代码就是普通文本等基础知识

不了解的，可以先去看：

[【整理】计算机语言基础知识介绍](#)¹⁵

以了解关于计算机语言的宏观介绍；

再看：

[【整理】Python语言简介](#)¹⁶

就明白了。

- 已经在Windows中安装好了Python
关于此部分内容，详见：

[第3章 如何安装Python](#)

另外提示一下，关于版本的选择：

[【整理】总结Python2\(Python 2.x版本\)和Python3\(Python 3.x版本\)之间的区别](#)¹⁷

下面，通过最简单的Python代码,此处只是打印一些Python版本信息和系统信息：

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
"""
Function:
【整理】如何在Windows下开发Python（如何运行Python脚本）

http://www.crifan.com/how_to_do_python_development_under_windows_environment

Author: Crifan Li
Version: 2012-12-06
"""

import platform;

pythonVersion = platform.python_version();
uname = platform.uname();
```

¹⁵ http://www.crifan.com/computer_language_basic_knowledge_introduction

¹⁶ http://www.crifan.com/simple_intro_what_is_python_and_how_to_run_python_script

¹⁷ http://www.crifan.com/summary_the_difference_between_python2_and_python3

```
print "Just for demo how to do python development under windows:";
print "Current python version info is %s"%(pythonVersion);
print "uname=",uname;
```

来说明，如何在Windows下，进行Python开发。

4.1.1. Python的最原始的开发方式是什么样的

相对来说，最原始的，最基本的，开发Python的方式，只是：

4.1.1.1. 找个文本编辑器，新建个.py文件，写上Python代码

Python代码，本身就只是文本；

所以，从原理上来说，只需要找个合适的文本编辑器，即可；

换句话说，如果只是简单的写写几行Python代码，你用Windows的记事本（Notepad），都是可以的；

只不过，由于Notepad功能太弱，所以在此个人推荐Notepad++。



为何推荐用Notepad++去写Python代码

此处，推荐用Notepad++去写Python代码，做Python开发，的原因

除了本身Notepad++作为文本编辑器，好用之外，

还有一个，相对来说，比较重要的问题：字符编码

很多时候，由于Python开发者对于Python文件的编码，不太了解

导致在开发期间，出现很多，相对比较常见的字符编码的问题

所以，为了，更加深入的了解和学习，真正的掌握Python

有必要，从一开始学习Python的时候

就选用合适的工具，这样，避免后期，一些该知道的细节，尤其是文件的编码，不知道而犯各种常见的错误

而关于Notepad++本身，不会使用，不熟悉，则可参考：

[【crifan推荐】轻量级文本编辑器，Notepad最佳替代品：Notepad++](http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html)¹⁸

关于Notepad++的几乎一切的内容，都可以在上述的教程中找到。

此处呢，对应的就是：

用我所推荐的Notepad++，新建一个文件，然后包括代码进入，存为对应的一个.py的文件。

而关于如何操作，此处也一点点截图，透彻的说明一下：

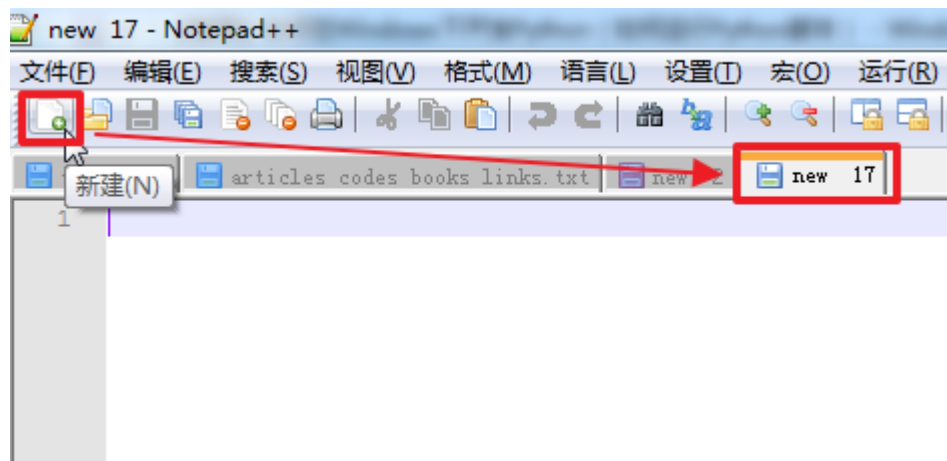
此处再次提示一下，一下Notepad++的所有功能，特性，用法，上面那个帖子中，都有专门的介绍。

¹⁸ http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html

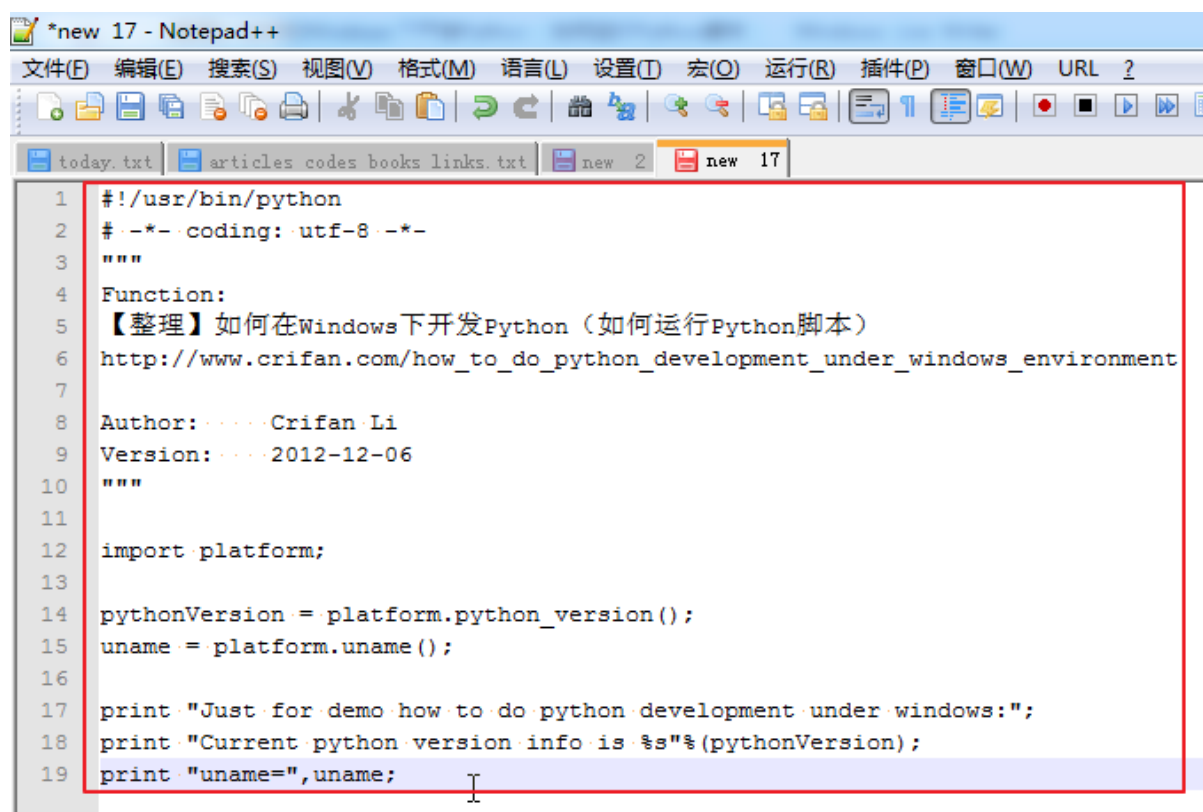
想要更加详细了解的相关的Notepad++的某个特定功能的用法的，自己去看即可。

下面的截图，就不再事无巨细的解释Notepad++的功能特点了。只是截图解释如何操作而已。

打开Notepad++后，点击新建，新建出一个新的文件：



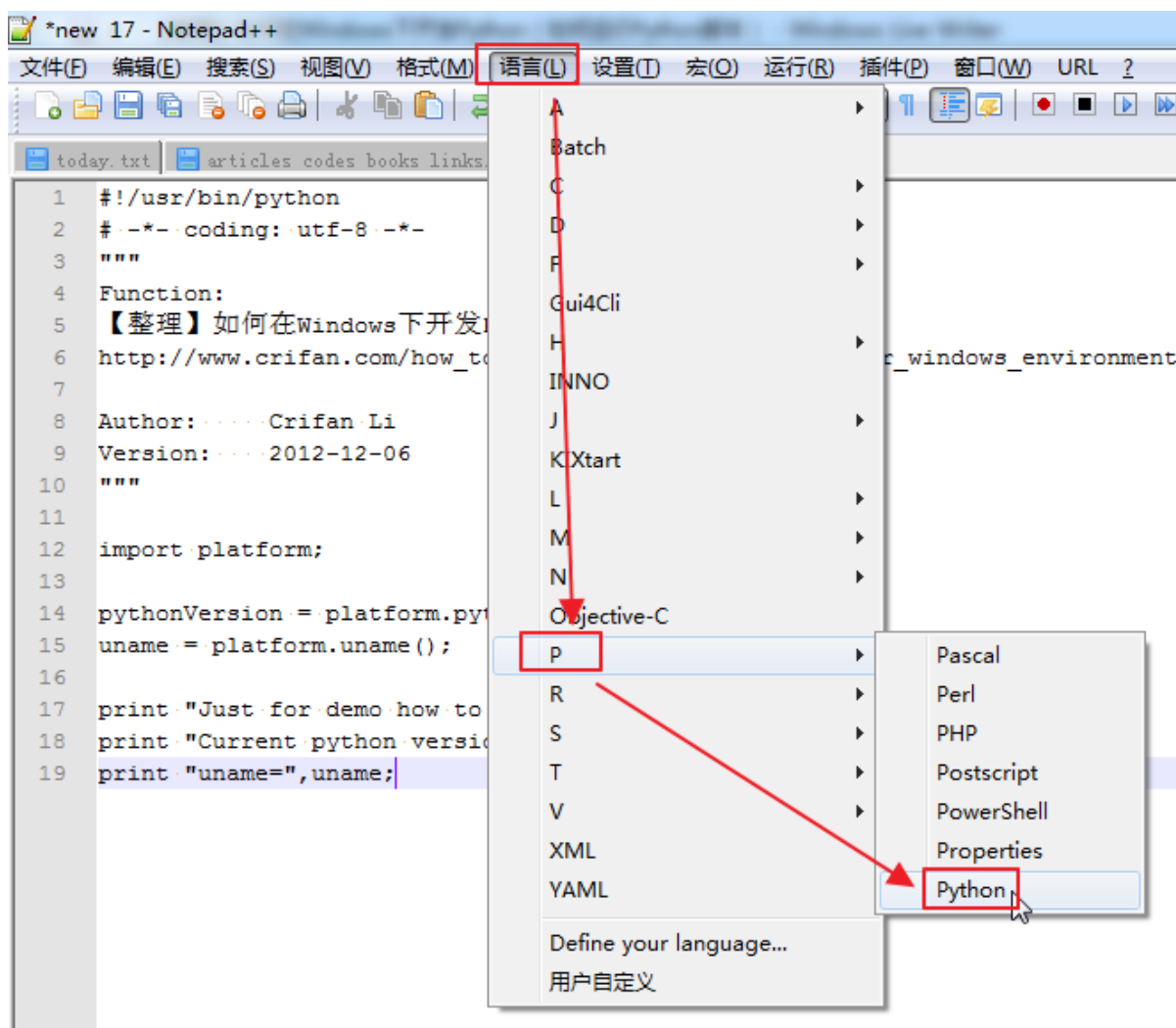
然后把上面给你们的代码，拷贝进去，就变成了：



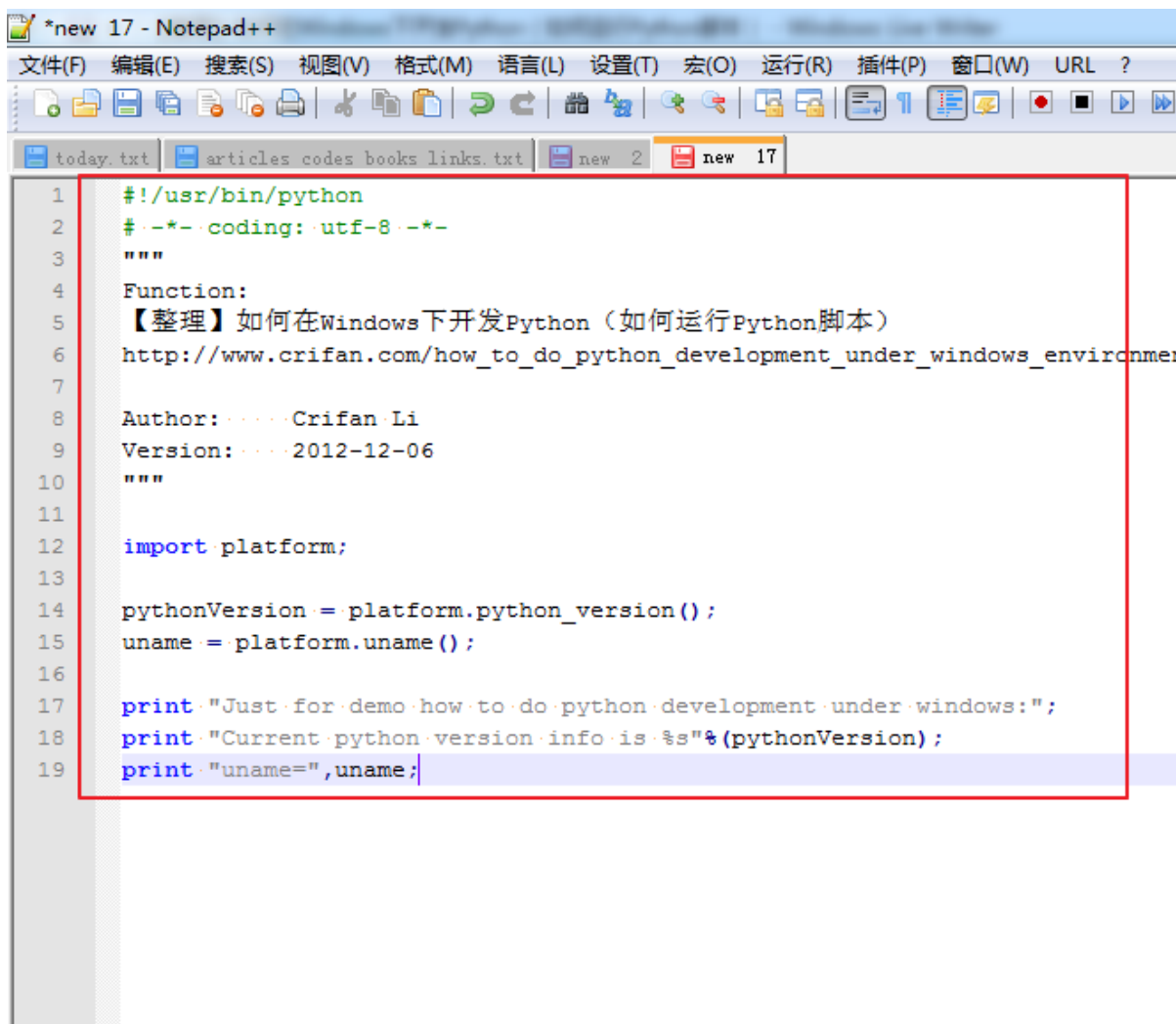
此处由于是新建的空白文件，所以Notepad++，不知道你是Python代码，没法帮你自动实现语法高亮，

需要手动去设置一下：

语言⇒P⇒Python



就可以看到Python代码的语法高亮的效果了：



```
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3  """
4  Function:
5  【整理】如何在Windows下开发Python（如何运行Python脚本）
6  http://www.crifan.com/how_to_do_python_development_under_windows_environment
7
8  Author: . . . . Crifan Li
9  Version: . . . . 2012-12-06
10 """
11
12 import platform;
13
14 pythonVersion = platform.python_version();
15 uname = platform.uname();
16
17 print "Just for demo how to do python development under windows:";
18 print "Current python version info is %s"%(pythonVersion);
19 print "uname=",uname;
```

然后去保存到某个位置。

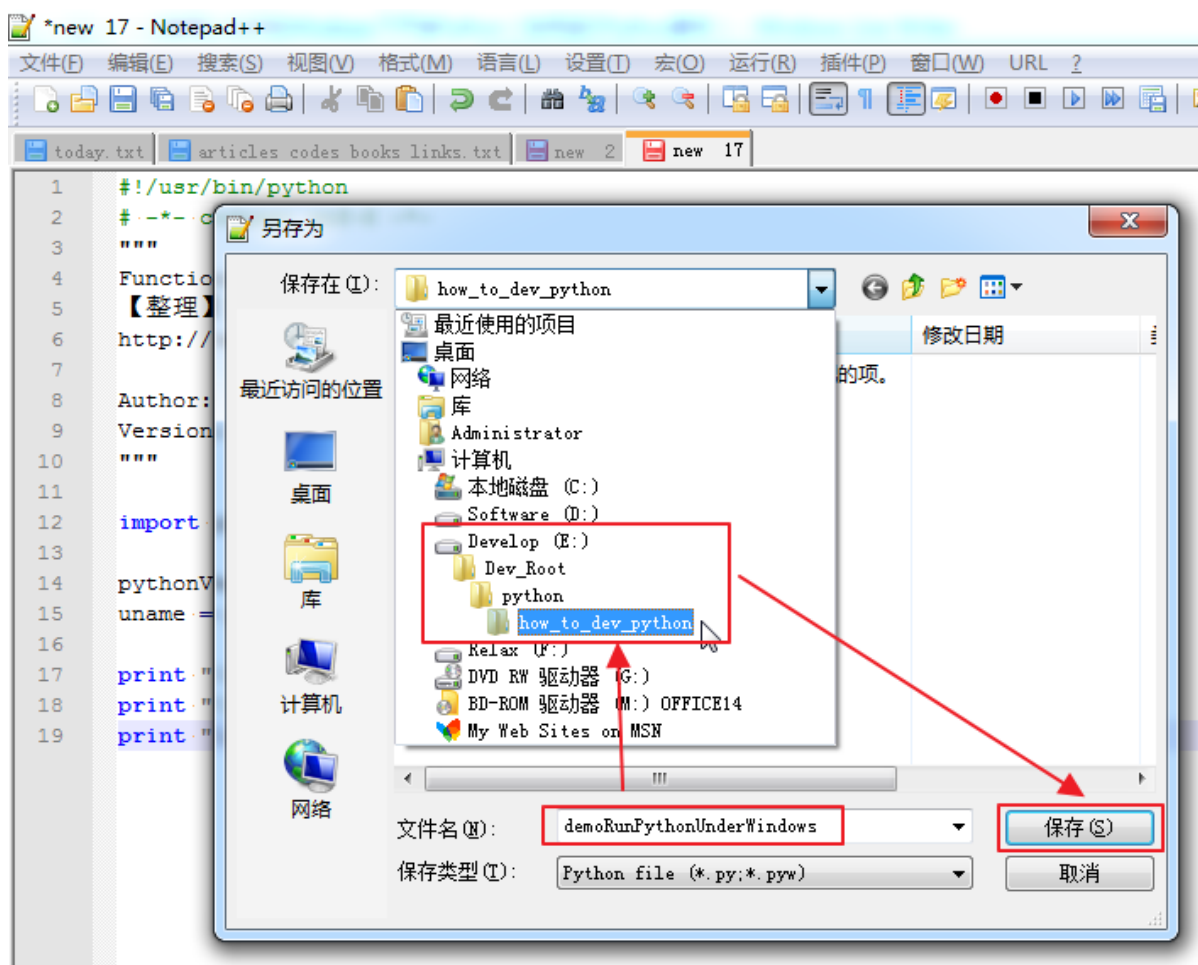
文件⇒另存为

在弹出的对话框中，输入要保存的文件名：demoRunPythonUnderWindows

提示：其中可以看到[Notepad++](http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html)¹⁹自动帮你写好了.py后缀，那是因为你之前设置了Python语法高亮。

然后再选择对应的路径去保存：

¹⁹ http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html



路径中不要包含（中文，空格等）特殊字符

对于保存文件来说，需要注意的是：

Python，以及其他语言，开发期间，最好都不要让路径中带有特殊字符

此处所谓的特殊字符，指的是非（ASCII）英文字符，主要包括：

- 中文字符
- 空格
- 其他特殊字符

而只包含普通的英文字符，即字母数字下划线。

比如，我此处的路径，用的是：

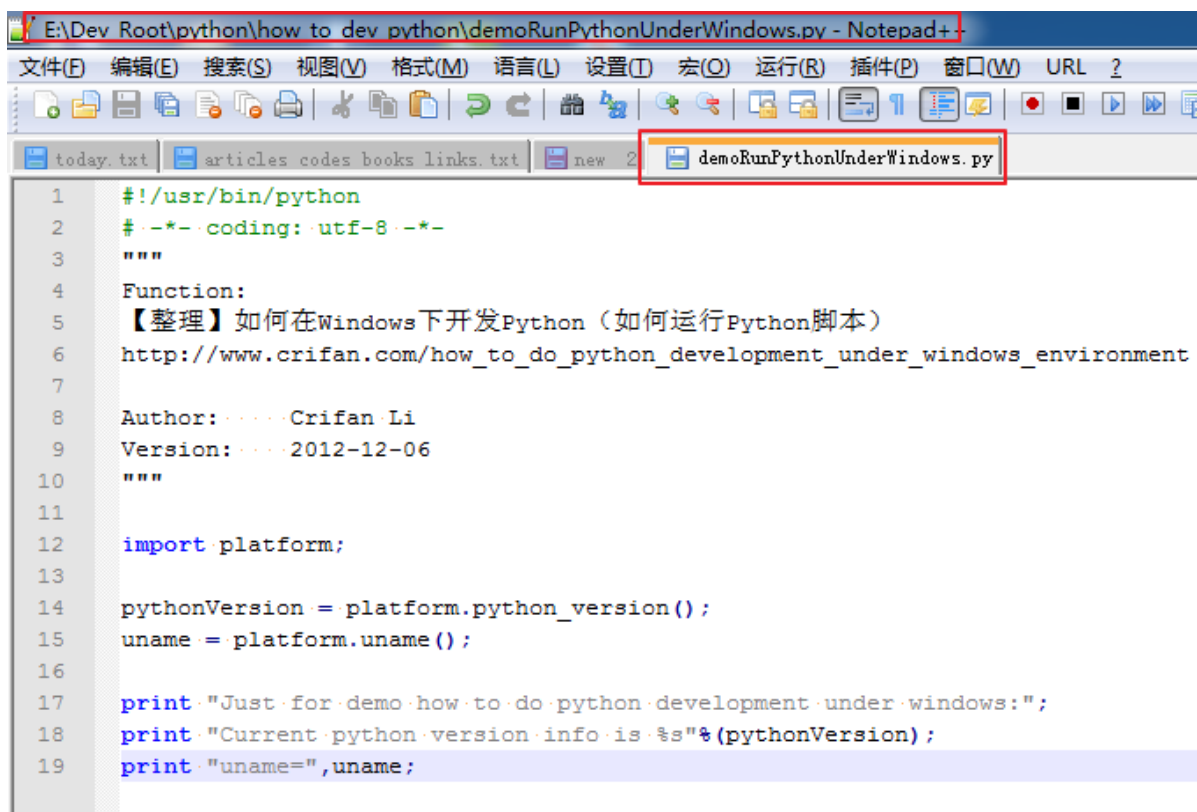
E:\Dev_Root\python\how_to_dev_python\

否则，很容易在后期开发期间，出现一些异常现象

比如找不到文件，文件夹等问题

以及常见的中文路径无法识别等问题。

保存好后，就是这样的了：



```
1  #!/usr/bin/python
2  #-*-coding:utf-8 -*-
3  """
4  Function:
5  【整理】如何在Windows下开发Python（如何运行Python脚本）
6  http://www.crifan.com/how_to_do_python_development_under_windows_environment
7
8  Author: .....Crifan·Li
9  Version: .....2012-12-06
10 """
11
12 import platform;
13
14 pythonVersion = platform.python_version();
15 uname = platform.uname();
16
17 print "Just for demo how to do python development under windows:";
18 print "Current python version info is %s"%(pythonVersion);
19 print "uname=",uname;
```

4.1.1.2. 打开Windows的cmd，并且切换到对应的python脚本所在目录

且换到对应的，Python文件所在的，文件夹，有两种办法：

4.1.1.2.1. 方法1：手动打开cmd，并cd到对应路径

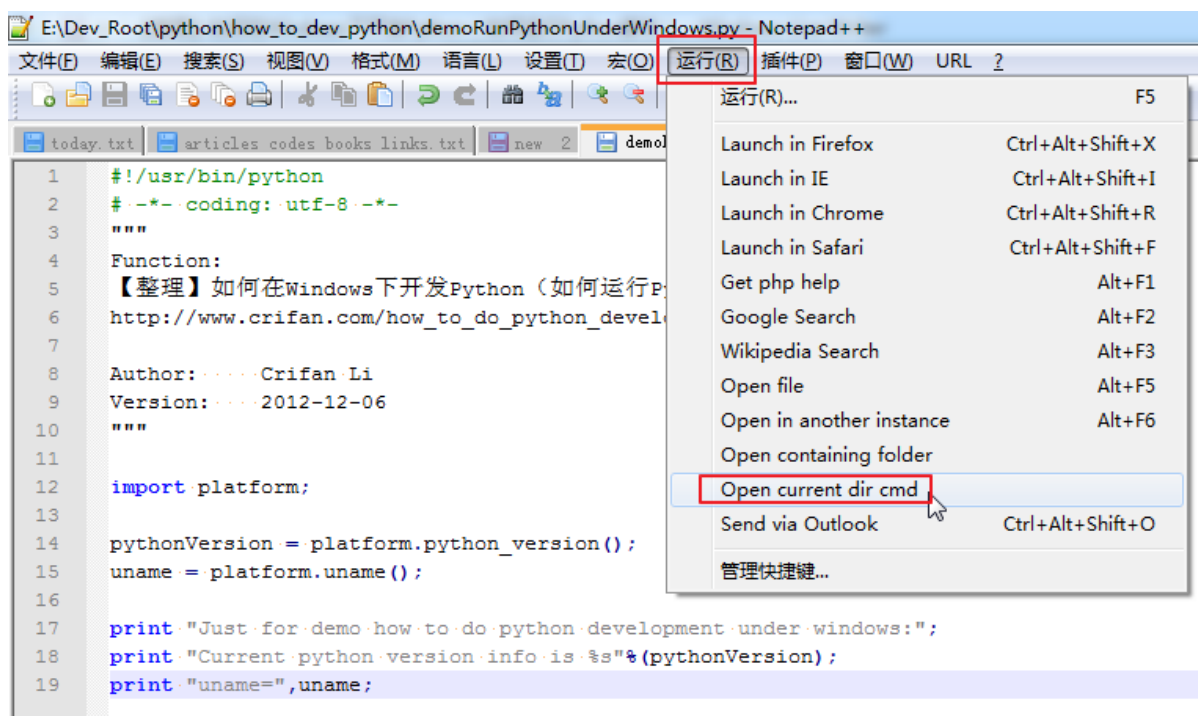
而对于，什么是windows的cmd，如何打开cmd，如何切换到对应的路径等内容，不熟悉的话，可以去参考：

[Windows的命令行工具: cmd](http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#win_cmd) ²⁰

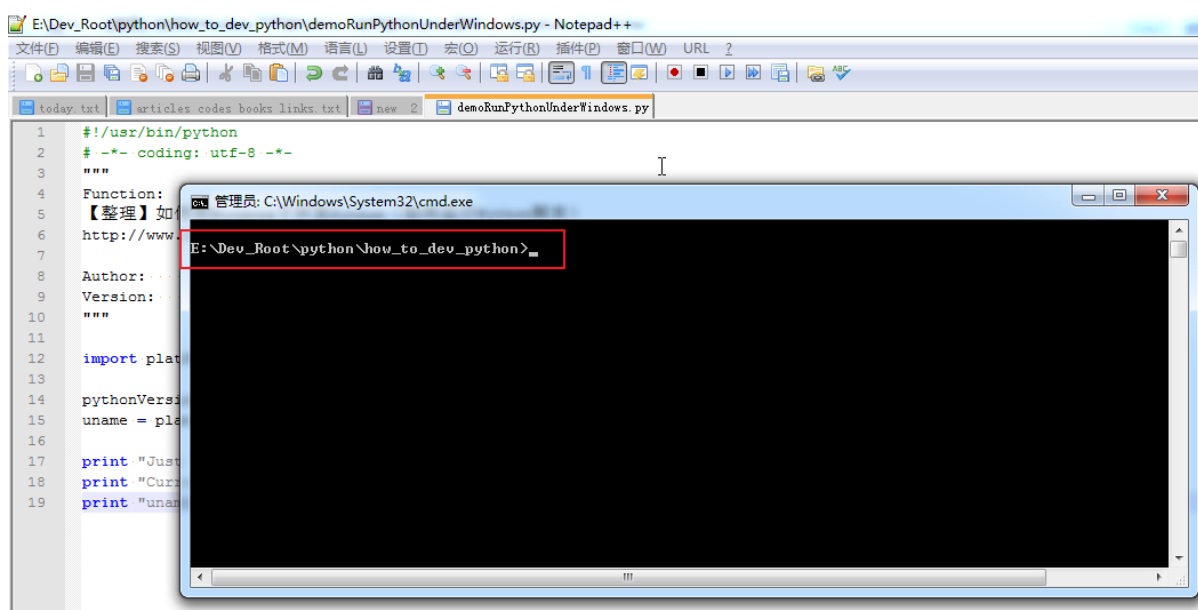
4.1.1.2.2. 方法2：通过Notepad++的Open current dir cmd

此处，使用一个更方便的办法，利用Notepad++中功能，直接打开cmd，并切换到对应路径：

²⁰ http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#win_cmd



然后就直接实现，打开了cmd，并且切换到对应的当前文件所在路径了：



4.1.1.3. 在cmd中去运行你的Python脚本（.py文件）

然后就是，在cmd中，输入你的Python脚本，即.py文件的完整的文件名。

此处是：

demoRunPythonUnderWindows.py



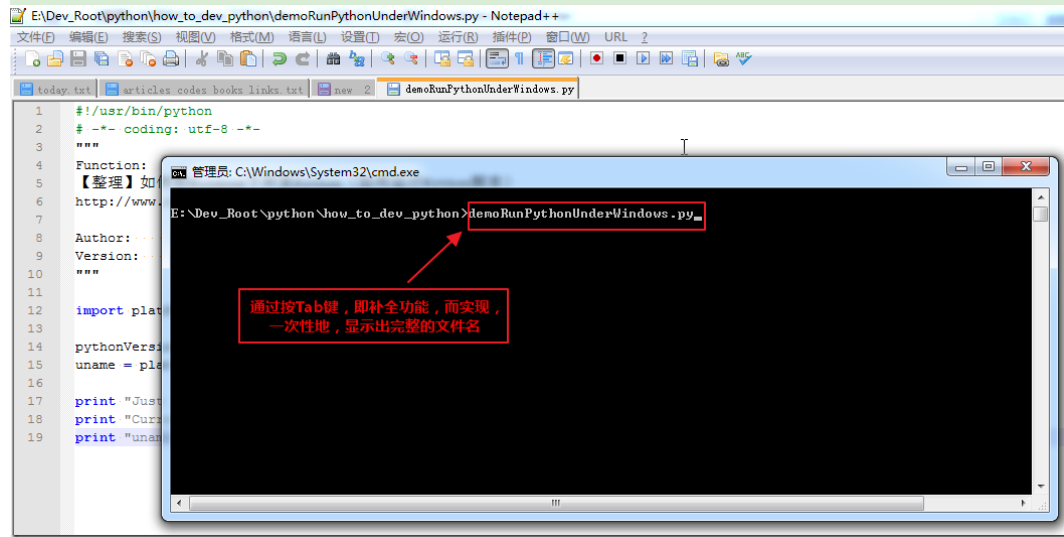
利用TAB键实现自动输入

正常的是通过手动输入对应的Python的完整的文件名，即，对于此处的demoRunPythonUnderWindows.py，一个个字母的输入

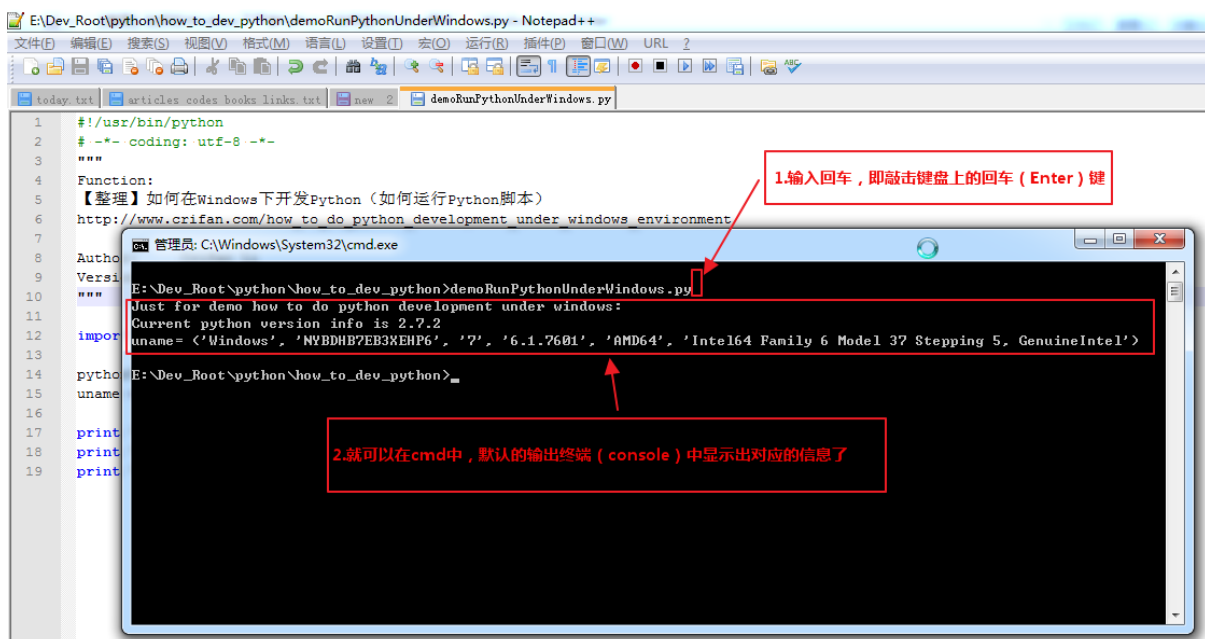
不过，此处有个小技巧，可以利用Windows（和Linux中通用的）**Tab**键，然后会自动显示出你当前目录的文件；

如果当前文件夹有多个文件，多次按**Tab**键，会在多个文件之间切换。

此处就一个文件，所以通过按**Tab**键，就可以一下子就显示出对应的整个文件名了：



然后输入回车，即可运行对应的Python脚本，接着在cmd中也就可以看到输出的结果了：



如此，就是一个，完整的，在windows的cmd中，运行Python脚本的流程了。

而接下来，作为正常的开发Python的流程就只是：

- 你继续去修改你的Python代码，添加新的代码，然后保存python文件，
- 再回到cmd中，重新再次运行Python，以验证程序运行是否正常，是否获得了你所期望的结果。
- 如此反复，一点点，由少到多，一点点写出足够复杂的Python代码，实现你的复杂的功能。

4.1.2. 利用Python的shell进行交互式开发又是怎样的

Python有个shell，提供一个Python运行环境。方便你交互式开发。

即写一行代码，就可以立刻被运行，然后方便查看到结果。

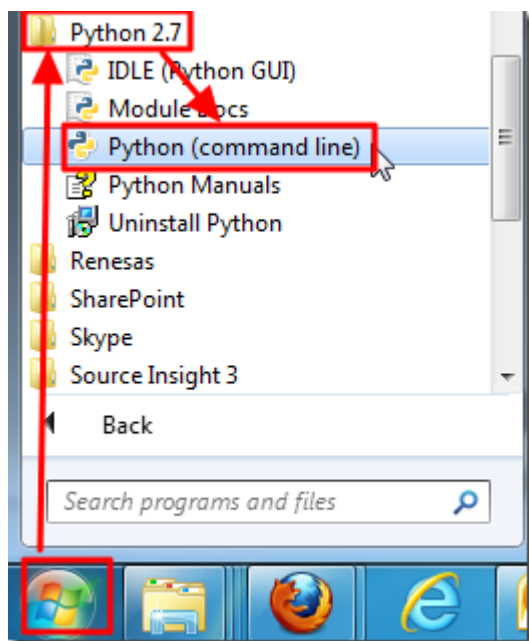
而Python的Shell，在Windows环境下，又分两种：

- Python (command line)
- IDLE (Python GUI)

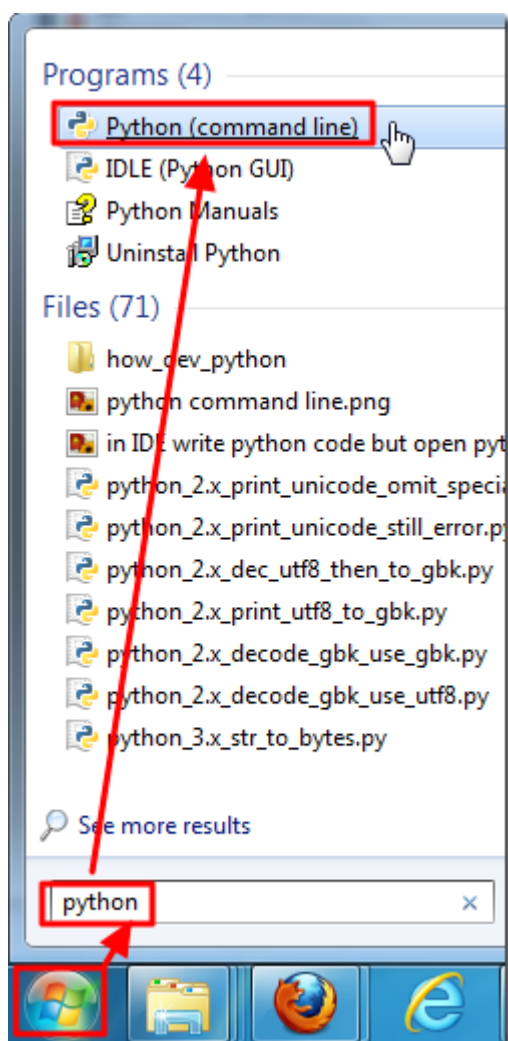
下面分别介绍一下。

4.1.2.1. 命令行版本的Python Shell – Python (command line)

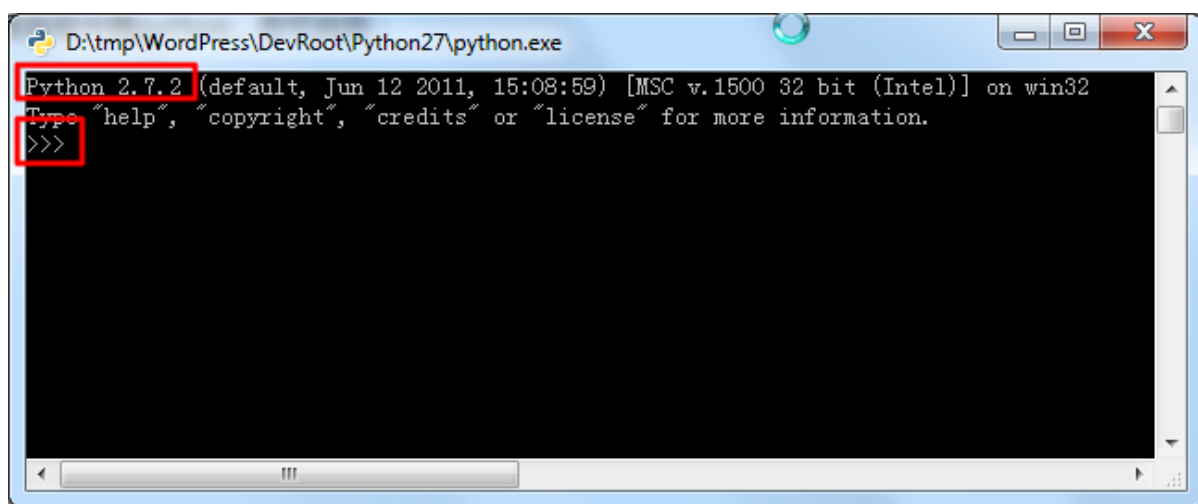
Windows下，安装好Python后，你可以在开始菜单中，找到对应的command line版本的Python Shell的：



其实，Win7中，有个更方便的方式，直接在搜索框中搜python，即可找到：



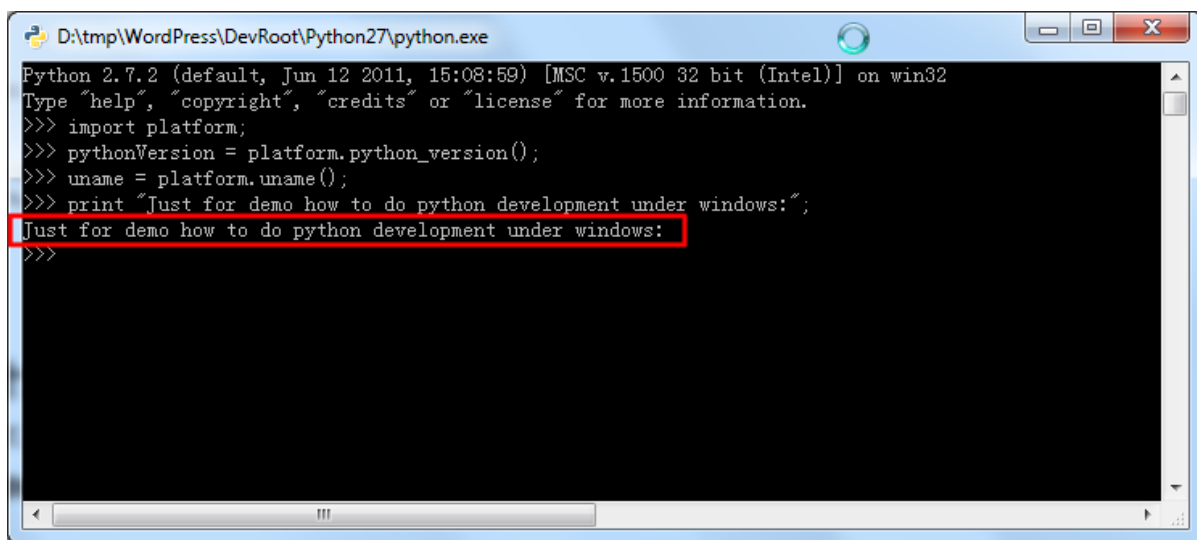
打开后，就是这个样子的：



其中可以显示出来对应的python版本信息和系统信息。

然后就是三个大于号>>>接下来，你就可以像在之前所说的，

普通文本中输入python代码一样，在此一行行输入代码，然后就可以显示对应的信息了：



```
D:\tmp\WordPress\DevRoot\Python27\python.exe
Python 2.7.2 (default, Jun 12 2011, 15:08:59) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import platform;
>>> pythonVersion = platform.python_version();
>>> uname = platform.uname();
>>> print "Just for demo how to do python development under windows:";
Just for demo how to do python development under windows:
>>>
```



为何叫做交互式shell (interactive shell)

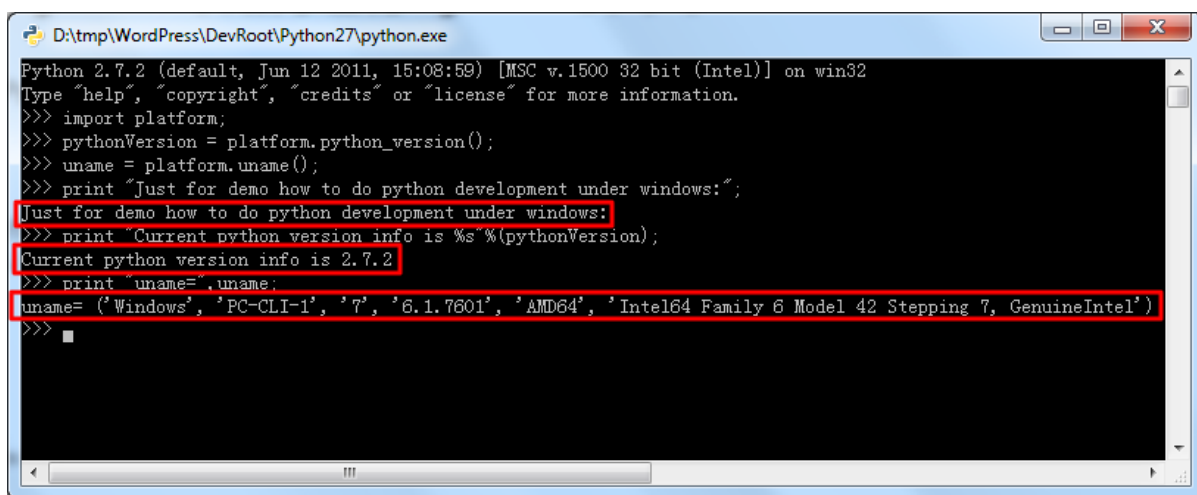
其中可以看到，当你输入对应的，第一行print时，此处命令行 (command line) 版本的Python Shell中，

就可以，动态的，交互式地，显示出对应的信息了。

正由于，此处可以，直接地，动态的，交互性地，显示出对应的信息，

所以，才被叫做Python 的交互式的Shell，简称Python Shell。

对应的，把前面的代码都输入完毕，结果显示为：

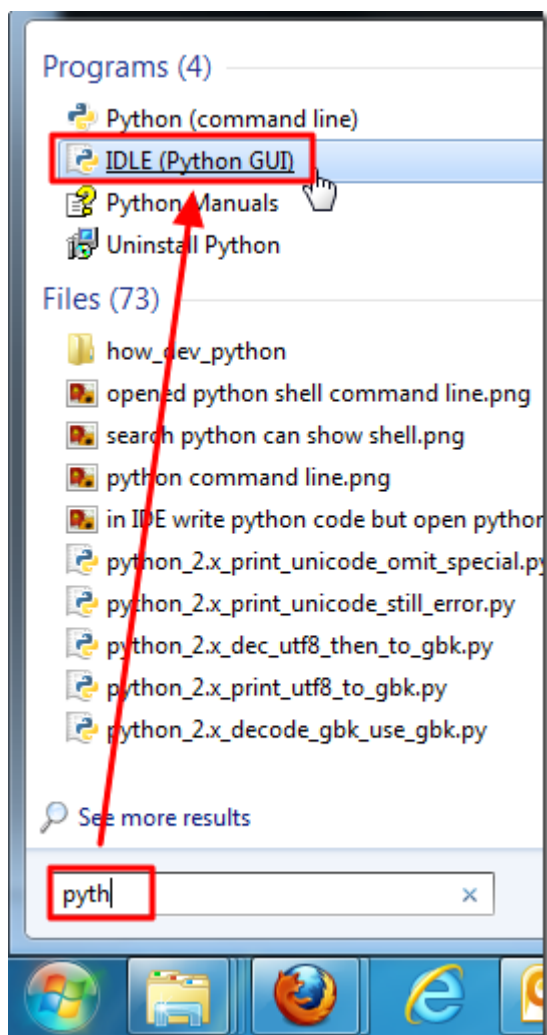


```
D:\tmp\WordPress\DevRoot\Python27\python.exe
Python 2.7.2 (default, Jun 12 2011, 15:08:59) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import platform;
>>> pythonVersion = platform.python_version();
>>> uname = platform.uname();
>>> print "Just for demo how to do python development under windows:";
Just for demo how to do python development under windows:
>>> print "Current python version info is %s"%(pythonVersion);
Current python version info is 2.7.2
>>> print "uname=",uname;
uname= ('Windows', 'PC-CLI-1', '7', '6.1.7601', 'AMD64', 'Intel64 Family 6 Model 42 Stepping 7, GenuineIntel')
>>>
```

4.1.2.2. 带图形界面的Python Shell – IDLE (Python GUI)

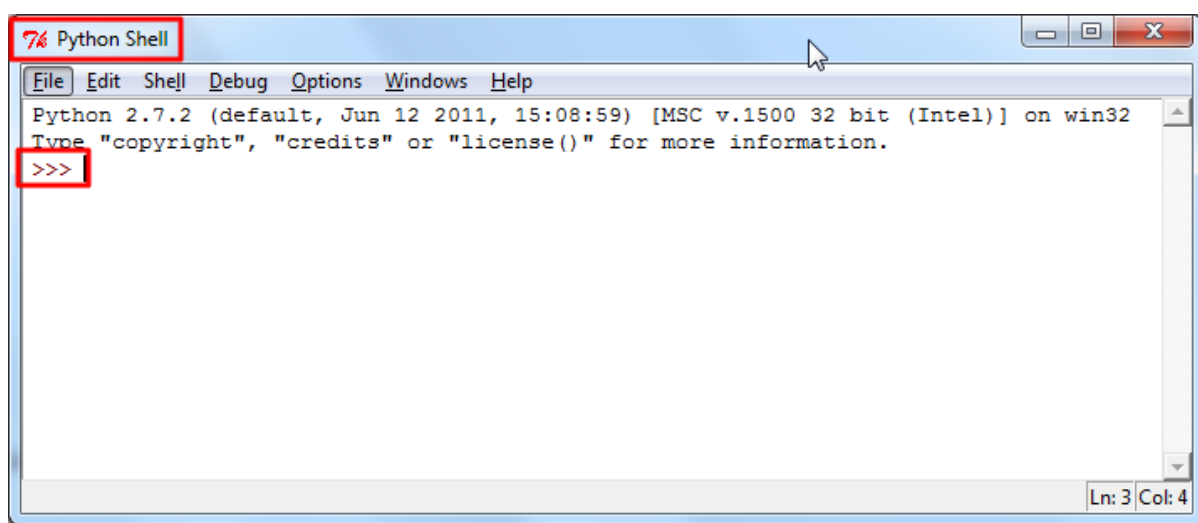
下面将要介绍的，同样是Python的Shell，但是是相对于command line版本而言，是GUI，带图形界面的版本的。

对应的打开方式，和上面类似，可以直接搜python而找到：



对应的，可以看到，其名字写的是IDLE

打开后就是这样的：



IDLE左上角的红色TK图标表示IDLE是用Tkinter图形库开发的

左上角的红色图标，好像是TK，是图形界面库的一种。

这个就是，很多Python教程中，

所常用来作为Python的开发环境，教别人写Python代码的那个IDLE。

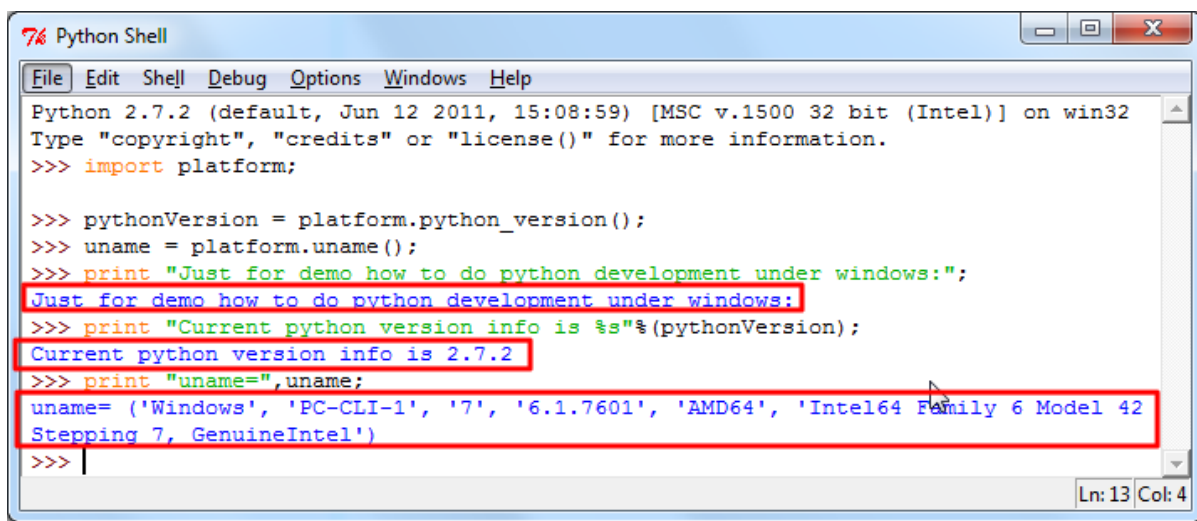
关于Python中的Tkinter图形库，详见：

[【整理】Python中的图形库](#)²¹

中的：

[【记录】折腾Python中的Tkinter](#)²²

对应的，输入上述的代码，结果也是类似的：



```
Python Shell
File Edit Shell Debug Options Windows Help
Python 2.7.2 (default, Jun 12 2011, 15:08:59) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import platform;

>>> pythonVersion = platform.python_version();
>>> uname = platform.uname();
>>> print "Just for demo how to do python development under windows:"
Just for demo how to do python development under windows:
>>> print "Current python version info is %s"%(pythonVersion);
Current python version info is 2.7.2
>>> print "uname=",uname;
uname= ('Windows', 'PC-CLI-1', '7', '6.1.7601', 'AMD64', 'Intel64 Family 6 Model 42
Stepping 7, GenuineIntel')
>>>
```



IDLE中一次性粘贴多行代码再运行则会出错

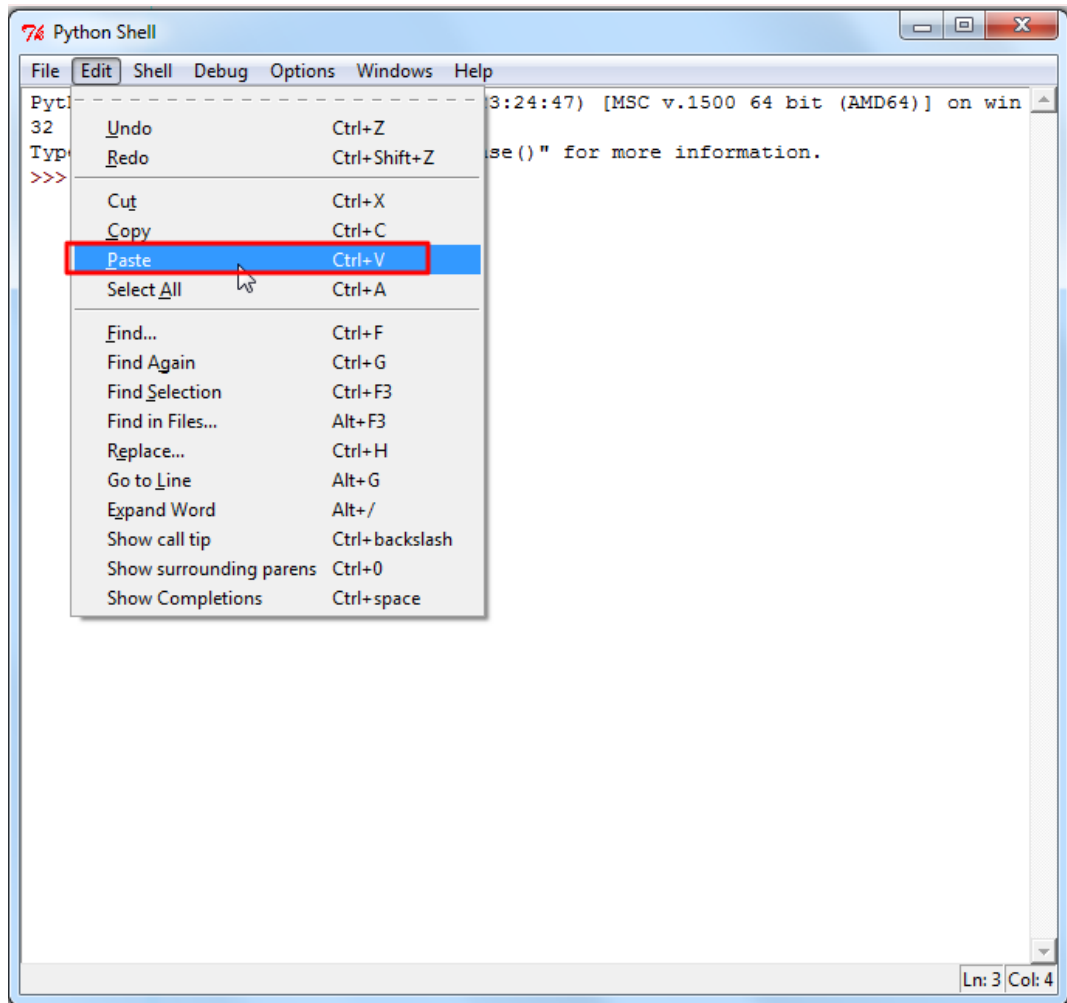
如果你对于IDLE的运行方式：交互式的，一行一行的输入，才能正常运行

不熟悉的话，作为新手，有些人会犯这样的错误：

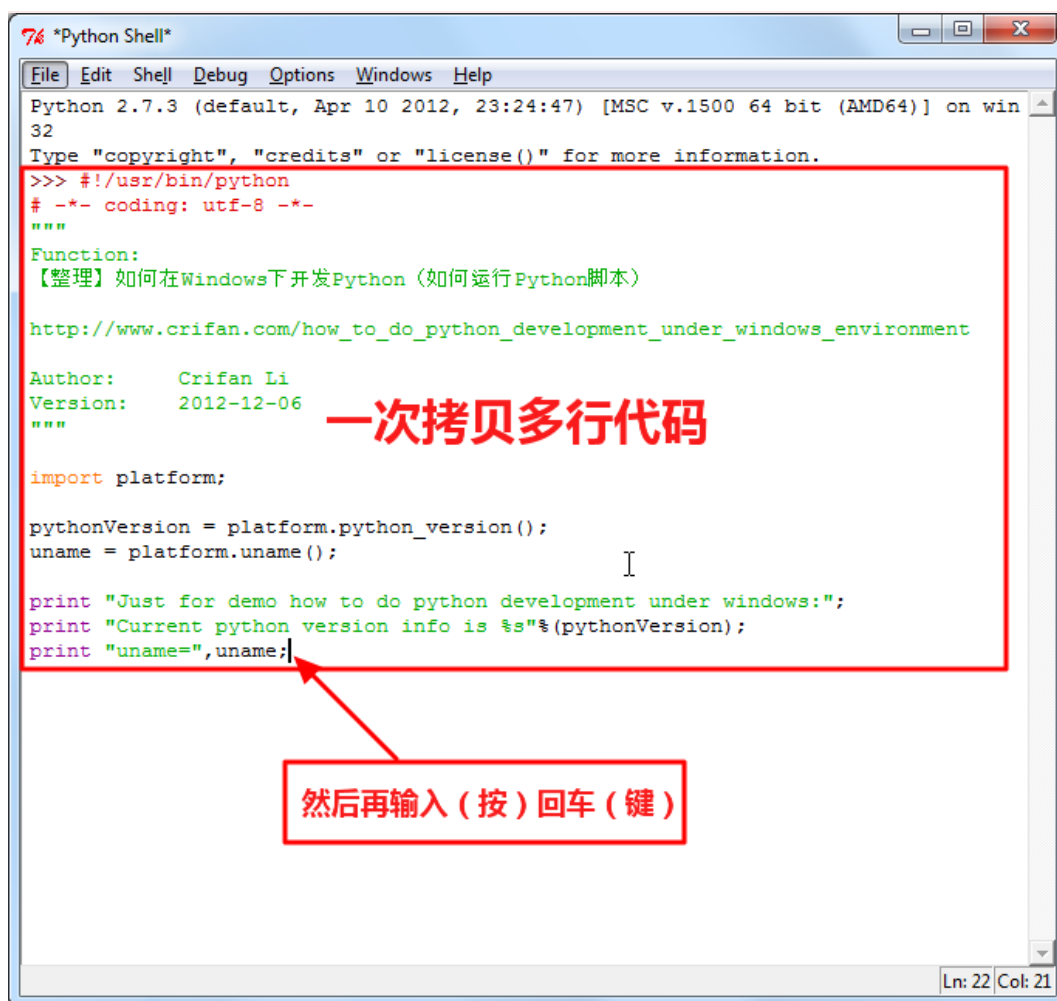
在别处拷贝Python示例代码，然后一次性的粘贴多行代码到IDLE中

²¹ http://www.crifan.com/summary_python_graphics_gui_libs_packages/

²² http://www.crifan.com/try_python_tkinter_module/



然后输入回车去运行：



7% *Python Shell*

File Edit Shell Debug Options Windows Help

Python 2.7.3 (default, Apr 10 2012, 23:24:47) [MSC v.1500 64 bit (AMD64)] on win32

Type "copyright", "credits" or "license()" for more information.

```
>>> #!/usr/bin/python
# -*- coding: utf-8 -*-
"""
Function:
【整理】如何在Windows下开发Python (如何运行Python脚本)

http://www.crifan.com/how_to_do_python_development_under_windows_environment

Author:    Crifan Li
Version:   2012-12-06
"""

import platform;

pythonVersion = platform.python_version();
uname = platform.uname();

print "Just for demo how to do python development under windows:";
print "Current python version info is %s"%(pythonVersion);
print "uname=",uname;
```

一次拷贝多行代码

然后再输入 (按) 回车 (键)

Ln: 22 Col: 21

结果，很明显，会出错：

此处是不能正常输入，之前所期望的python版本信息，

而是出现其他异常情况：只是显示此处的注释代码

（因为此处最开始部分的代码，就只是注释代码）

（如果你是其他的多行代码，则对应的就是出现其他对应的错误了）



```
Python Shell
File Edit Shell Debug Options Windows Help
Python 2.7.3 (default, Apr 10 2012, 23:24:47) [MSC v.1500 64 bit (AMD64)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> #!/usr/bin/python
# -*- coding: utf-8 -*-
"""
Function:
【整理】如何在Windows下开发Python (如何运行Python脚本)

http://www.crifan.com/how_to_do_python_development_under_windows_environment

Author:      Crifan Li
Version:     2012-12-06
"""

import platform;

pythonVersion = platform.python_version();
uname = platform.uname();

print "Just for demo how to do python development under windows:";
print "Current python version info is %s"%(pythonVersion);
print "uname=",uname;
'\nFunction:\n\n\x1\xbe\xd5\xfb\x0\xed\x1\xbf\x08\xe7\xba\xce\xd4\xdaWindows\xcf\x02\xbf\xaa\xb7\xa2Python\xa3\xa8\x08\xe7\xba\xce\xd4\xcb\xd0\xd0Python\xbd\x05\x01\xbe\xa3\xa9\n \nhttp://www.crifan.com/how_to_do_python_development_under_w
indows_environment\n \nAuthor:      Crifan Li\nVersion:     2012-12-06\n'
>>> |
```

Idle中无法执行你所，一次性拷贝的多行代码，所以此处只是
显示出一些错误（不正常的信息，非正常代码执行的输出信息）

而，如果想要正确的运行代码，

则应该和之前一样，正常的，一行一行的输入代码，并回车，去执行

才能正常的，显示出你所希望看到的信息

正常的，一行一行的，输入代码，才能正确执行所输入的代码，显示出对应的输入结果

```

Python 2.7.3 (default, Apr 10 2012, 23:24:47) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> #!/usr/bin/python
# -*- coding: utf-8 -*-
"""
Function:
【整理】如何在Windows下开发Python (如何运行Python脚本)

http://www.crifan.com/how_to_do_python_development_under_windows_environment

Author:    Crifan Li
Version:   2012-12-06
"""

import platform;

pythonVersion = platform.python_version();
uname = platform.uname();

print "Just for demo how to do python development under windows:";
print "Current python version info is %s"%(pythonVersion);
print "uname=",uname;
'\nFunction:\n\xa1\xbe\xd5\xfb\xc0\xda\xa1\xbf\xc8\xe7\xba\xce\xda\xdaWindows\xcf\xca2\xbf\xaa\xb7\xa2Python\xa3\x
\xa8\xce7\xba\xce\xda\xcb\xda0Python\xbd\xce5\xbl\xbe\xa3\xa9\n \nhttp://www.crifan.com/how_to_do_python_dev
elopment_under_windows_environment\n \nAuthor:    Crifan Li\nVersion:   2012-12-06\n'
>>> import platform;
>>> pythonVersion = platform.python_version();
>>> uname = platform.uname();
>>> print "Just for demo how to do python development under windows:";
Just for demo how to do python development under windows:
>>> print "Current python version info is %s"%(pythonVersion);
Current python version info is 2.7.3
>>> print "uname=",uname;
uname= ('Windows', 'PC-CLI-1', '7', '6.1.7601', 'AMD64', 'Intel64 Family 6 Model 42 Stepping 7, GenuineIntel')
>>>

```

4.1.2.3. 关于 (command line或GUI版本的) Python Shell的用途

而作为一般的Python开发的话，则很少有用这个Python Shell的，不论是command line的还是GUI版本的IDLE。

其比较适合用来测试，演示一些简单的代码的执行的效果。

好处是很方便，可以立刻看到代码执行的结果。

所以，结论就是：

对于python的shell，不论是command line版还是GUI版，都比较适合偶尔要测试少量的Python代码的情况下去使用，而不适合长期的开发Python。

例 4.1. 举例：用Python的IDLE去做URL解码

比如我之前就是借用Python的一些库函数，实现一些对于url解码的功能：

【已解决】在用google搜索出来的链接无法打开的情况下，如何找到该链接的真实地址²³

4.1.3. 利用第三方Python的IDE进行Python开发又是怎么回事



什么是IDE

关于IDE的基本概念，不了解的先去看：

【整理】什么是IDE²⁴

²³ http://www.crifan.com/find_real_link_from_google_link_when_failed_open_via_google/

²⁴ http://www.crifan.com/what_is_ide

即：

公式 4.1. 什么是IDE

IDE

= 集成开发环境

= 把开发相关的各种环境（和工具）都集成到一起

而Python的IDE，就是：

公式 4.2. 什么是Python的IDE

Python IDE

= Python的集成开发环境

= 把和Python开发相关的各种工具

- Python代码编辑器：替代你前面用的Notepad++等文本编辑器
- Python的运行环境：模拟或替代：Python的（命令行或GUI版本的，交互式）shell

集成在一起

另外再加上各种文件，代码，项目的组织，管理等方面的各种功能

以此去方便你运行对应的Python代码，

方便你进行Python项目的开发

4.1.3.1. 为何会有Python的IDE

而之所以会有Python的IDE的诞生，也很容易理解。

就是因为，如果开发Python过程中，写Python代码，调试Python代码，查找相关的函数的解释等等操作，

如果都是基于前面介绍的，用Notepad++等编辑器去编辑Python代码，写完代码了，再切换到windows的cmd中去运行，

往往觉得很麻烦。

尤其是大型项目的话，可能就更加显得不那么高效；

以及对应的需要一些额外的功能，比如调试复杂的Python代码，需要一点点跟踪调试，找到错误的根本原因等等。

上述的开发模式，就更显得力不从心。

所以，才会有：

Python发展到现在，已经有了很多第三方的，别人开发的，可以用于或者专门用于Python开发的一些集成开发环境，即Python的IDE。

4.1.3.2. 目前常见的一些Python的IDE

参考别人的一些讨论和总结：

[【python】【求助】关于python编辑器的选择](#) ²⁵

[python编辑器对比和推荐](#) ²⁶

罗列几个，相对用的比较广泛的（排名不分先后）：

- Ulipad
- PyScripter
- Wing IDE
- Eclipse + pydev插件

关于其中的一些IDE的效果，可以参考：

[【整理】各种Python的IDE\(集成开发环境\)的总结和对比](#) ²⁷

4.1.3.3. Python的IDE和Python代码编辑器，Windows的cmd，等的关系

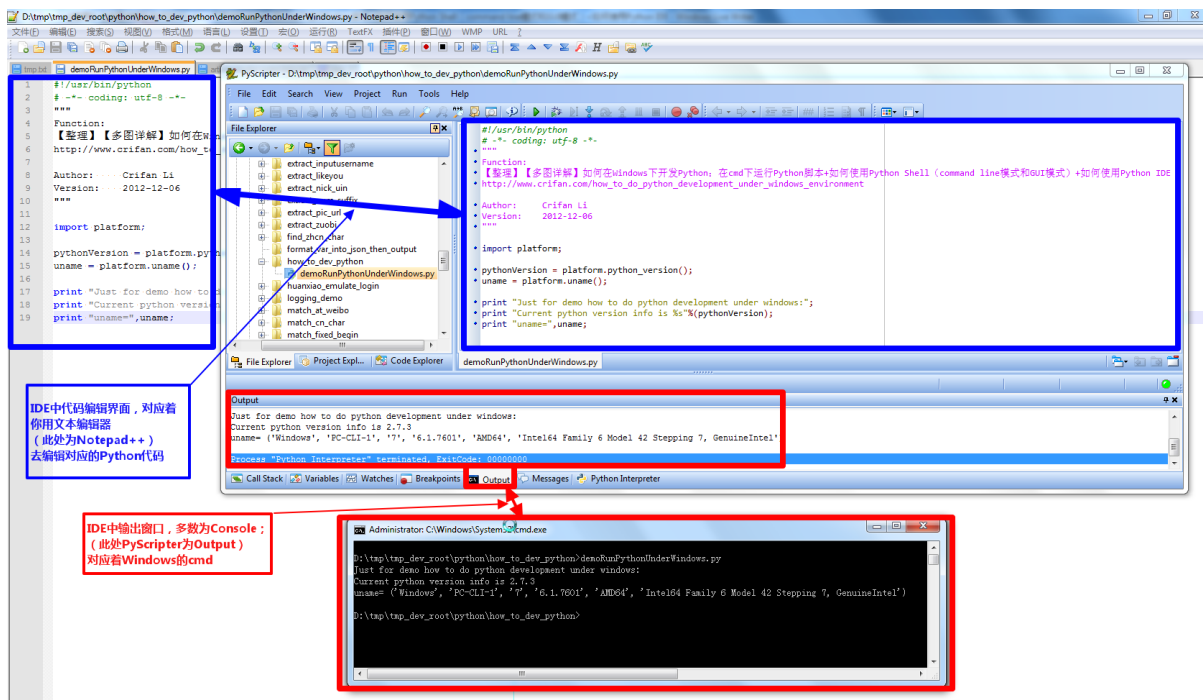
下面，随便找了一个Python的IDE，

比如

[【记录】使用Python的IDE：PyScripter](#) ²⁸

通过截图来对比性的解释，

这样你就更容易理解Python的原始开发环境和Python的IDE之间的关系了：

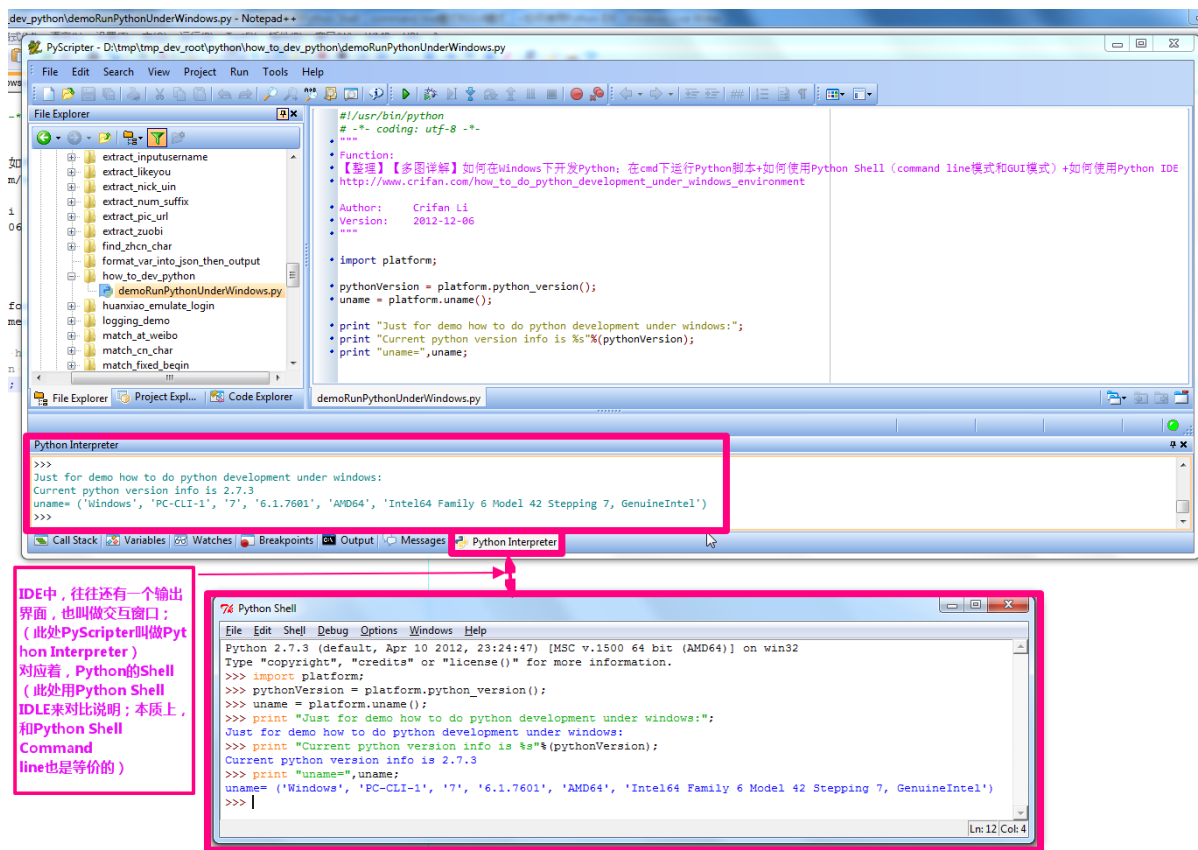


²⁵ <http://bbs.csdn.net/topics/390022660>

²⁶ <http://blog.csdn.net/cserchen/article/details/7036435>

²⁷ http://www.crifan.com/summary_common_python_ide_pyscripter_ulipad_eclipse_pydev_eric

²⁸ http://www.crifan.com/try_with_python_ide_pyscripter/



这下，至少你应该对于：

- Windows的cmd
- Python的Shell
 - command line版本
 - GUI版本：IDLE
- Python的IDE

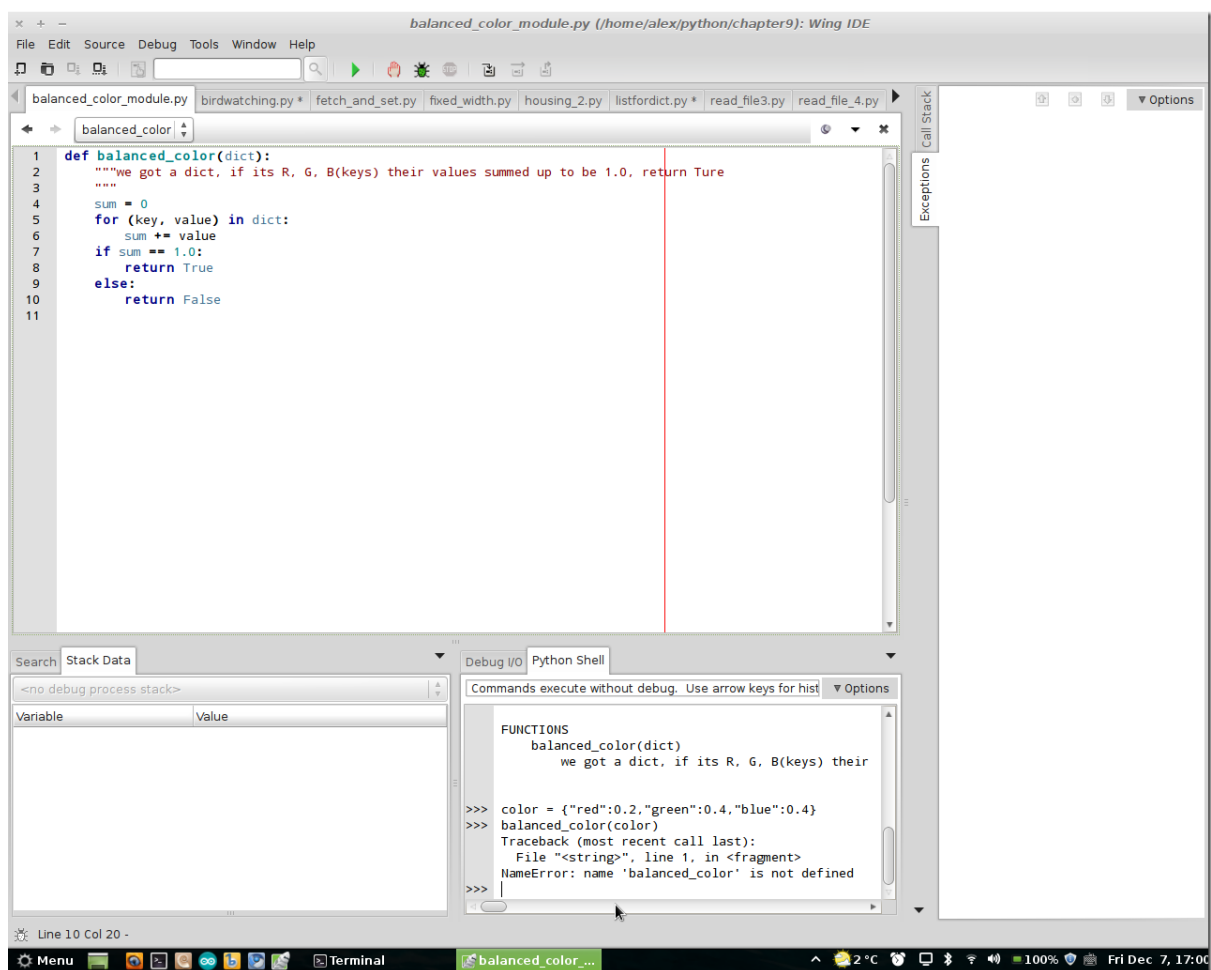
三者之间的关系，有个大概的认识了。

4.1.3.4. 使用IDE时所遇到的一些常见的问题

4.1.3.4.1. IDE只能够打开了文件，并不代表就已经在shell中运行了该文件

这里有一个错误的例子，比如[这里](http://zhidao.baidu.com/question/505232524.html)²⁹所遇到的：

²⁹ <http://zhidao.baidu.com/question/505232524.html>



如图，其在当前的IDE中，打开了一个.py文件：balanced_color_module.py

按照正常的逻辑，应该到菜单中去运行代码，然后再去对应的console中（此处估计是Debug I/O），去查看运行的结果。

但是，很明显，对于此处的那个py文件（balanced_color_module.py），即使运行该文件，也不会有什么print输出信息

另外，其此处没有打开对应的console，而是打开的是Python Shell，所以从逻辑上就是：

其在Python Shell中输入的任何代码，本质上和你当前IDE中所打开的py文件，都没有半毛钱关系的。

所以导致的结果就是，虽然在Python Shell中输入了一些代码：

```
color = {"red":0.2, "green":0.4, "blue":0.4}
balanced_color(color);
```

但是结果却出错，找不到对应的balanced_color函数。

所以，也就在常理之中，预料之内了。

因为，此时其所做的事情是：

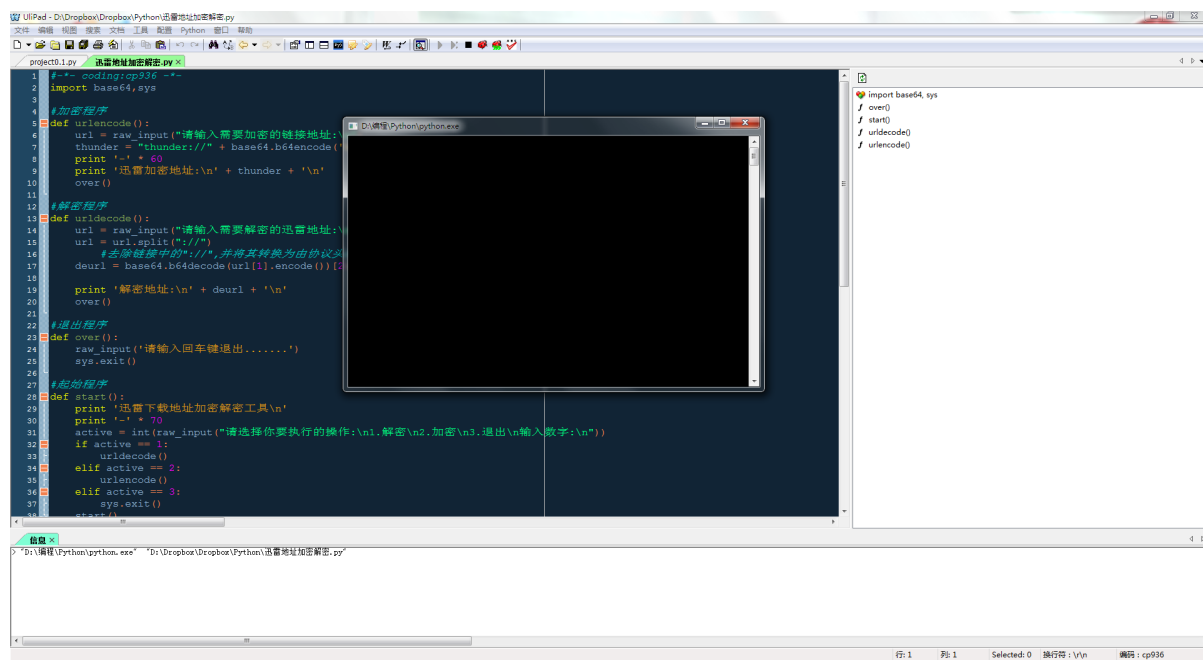
使用了IDE，但是并没有使用IDE中集成的终端调试输出

却另外打开了Python Shell，

而由于此处的IDE中打开的代码，和Python Shell没有半毛钱关系，所以才会运行代码出错，找不到对应的函数的。

4.1.3.4.2. 需要注意，确保有可以运行的Python起始部分的代码

比如[这里](#)³⁰遇到的问题，通过IDE中运行Python代码，但是没有响应：



其所遇到的问题，其实本质上，和IDE没有关系，但是不熟悉Python的人，还是容易在IDE中犯这类问题的。

错误原因是：

没有起始代码可以运行；

解决办法是：

把上面最后一行的

```
start();
```

去掉缩进，顶行写，变成：

```
start();
```

就可以了。使得Python可以执行到对应的start函数了。

当然，最好的写法是，把：

```
start();
```

再改为：

```
if __name__ == "__main__":
    start();
```

³⁰ <http://zhidao.baidu.com/question/505568675.html>

具体的解释，参见：

[【整理】Python中的__name__和__main__含义详解](#)³¹

4.1.4. 总结：到底使用哪种环境去开发Python

一句话：

各取所需。根据自己的需求，决定用什么开发环境。

4.1.4.1. 对初学者的建议：如何选用Python的开发环境

再加一句：

针对初学者，我个人倒是建议使用第一种，即windows的cmd下，去运行python脚本

目的很明确：

很多东西的学习，其本质上，都是需要一个循序渐进的过程的，学习Python语言同样如此。

在没有学会走路，即如何搞懂Windows的cmd下运行Python脚本，

就想学会跑了，即直接利用Python的IDE，包括shell和第三方开发环境，

结果就是，很多东西，还是不明白到底是为什么，理解的不透彻。

而当Python的基本知识，基本开发流程熟悉了之后，再建议你去使用第三方的Python的IDE，到时候，才能算是用着很爽。

即：

1. 先：[Windows的cmd](#)³² + [Notepad++](#)³³
2. 再：选用某个IDE³⁴，比如[PyScripter](#)³⁵，[Ulipad](#)³⁶，[Eclipse+PyDev](#)³⁷等。

4.1.5. 如何在Windows环境下使用Python脚本

首先要说明的是，据我目前的了解，对于一般Windows的用户来说，想要使用已有的Python脚本的话，主要有两种形式可以使用。

1. 直接运行文本式的Python脚本文件
我们所常见的，多数的Python脚本，都是此形式的。而关于Python脚本，其实就是一个文本文件，你可以用任何一个文本编辑器，比如windows的Notepad.exe来打开对应的后缀名为.py的文件，比如我所发布的，用于博客搬家到wordpress的Python脚本BlogsTo Wordpress.py

而对于在Windows的命令行，即cmd下运行Python脚本，用起来，就是这个样子的：

³¹ http://www.crifan.com/python_detailed_explain_about__name__and__main__/

³² http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#win_cmd

³³ http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html

³⁴ http://www.crifan.com/summary_common_python_ide_pyscripter_ulipad_eclipse_pydev_eric

³⁵ http://www.crifan.com/try_with_python_ide_pyscripter

³⁶ http://www.crifan.com/try_with_python_ide_ulipad

³⁷ http://www.crifan.com/try_with_python_ide_eclipse_pydev

图 4.1. 在Windows下的cmd下面运行Python脚本的样子

```

管理员: C:\Windows\system32\cmd.exe
E:\Dev_Root\svn_dev_root\website\python\BlogsToWordPress>BlogsToWordPress.py -s http://blog.sina.com.cn/lifecoaching
Imported: crifanLib, v1.6
Imported: BlogNetease, v1.4
Imported: BlogBaidu, v1.3
Imported: BlogSina, v1.4
Imported: BlogQQ, v1.7
Imported: BlogCsdn, v1.0
Imported: BlogSohu, v1.3
LINE 1139 : INFO 版本信息: v8.3
LINE 1140 : INFO 1.如发现bug, 请将日志文件和bug截图等信息发至: admin(at)crifan.com
LINE 1141 : INFO 2.如对此脚本使用有任何疑问, 请输入 -h 参数以获得相应的参数说明。
LINE 1142 : INFO 3.关于本程序详细的使用说明和更多相关信息, 请参考:
LINE 1144 : INFO BlogsToWordPress: 将百度空间, 网易163, 新浪Sina, QQ空间, 人人网, CSDN, 搜狐等
博客搬家到WordPress
LINE 1145 : INFO http://www.crifan.com/crifan_released_all/website/python/blogstowordpress/
LINE 191 : INFO -----
LINE 1186 : INFO Your process type of post is: Export post to WXR(WordPress eXtended Rss).
LINE 1230 : INFO Source URL: http://blog.sina.com.cn/lifecoaching
LINE 1379 : INFO Your blog provider : Sina Blog.
LINE 1086 : INFO Extracted Blog user [lifecoaching] from http://blog.sina.com.cn/lifecoaching
LINE 1087 : INFO Blog entry url is http://blog.sina.com.cn/lifecoaching
LINE 1065 : INFO Username and/or password is null, now in un-login mode.
LINE 1235 : INFO Now start to find the permanent link for http://blog.sina.com.cn/lifecoaching
LINE 1241 : ERROR Can not find the first link for http://blog.sina.com.cn/lifecoaching, error=Unk
nown error!
半:

```

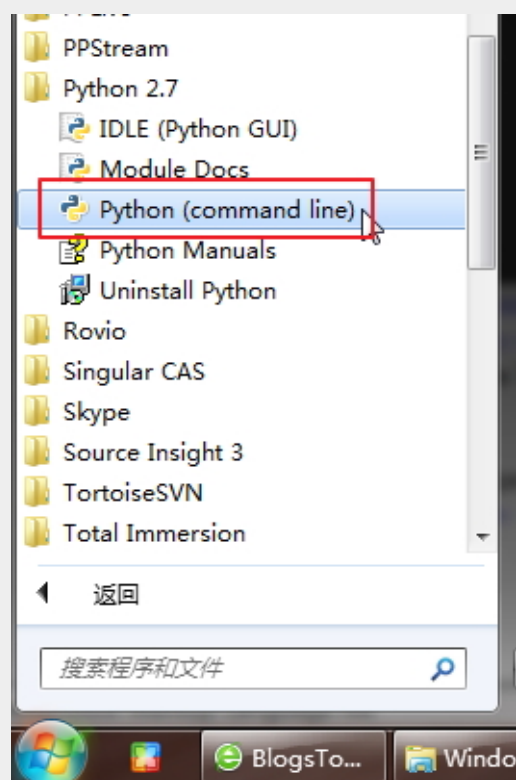


Windows的cmd下运行Python脚本，不是在Python的Command Line工具下运行Python脚本

对于此用法不太熟悉的新手，需要注意一点的是，不要把，在Windows下的cmd中运行Python脚本，和在Python（安装好后自带的）Command Line去写Python代码，运行Python代码，相混淆了

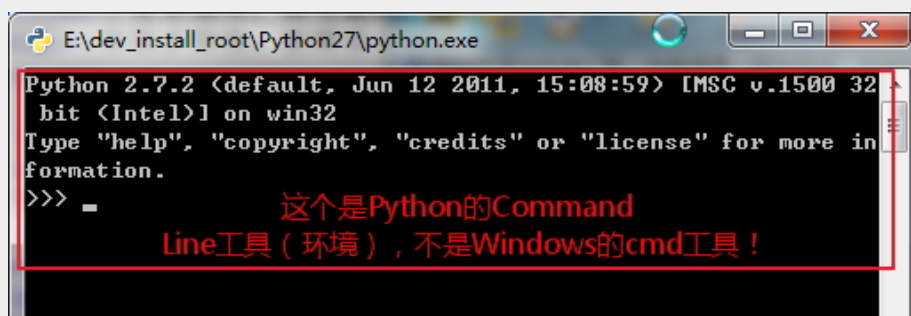
对于Python的Command Line，以我此处Python 2.7为例，是从开始菜单中找到的：开始菜单 ⇒ Python 2.7 ⇒ Python (Command Line)，如图：

图 4.2. 开始菜单中找到的Python (Command Line)



其打开后的效果如下：

图 4.3. Python (Command Line)的界面

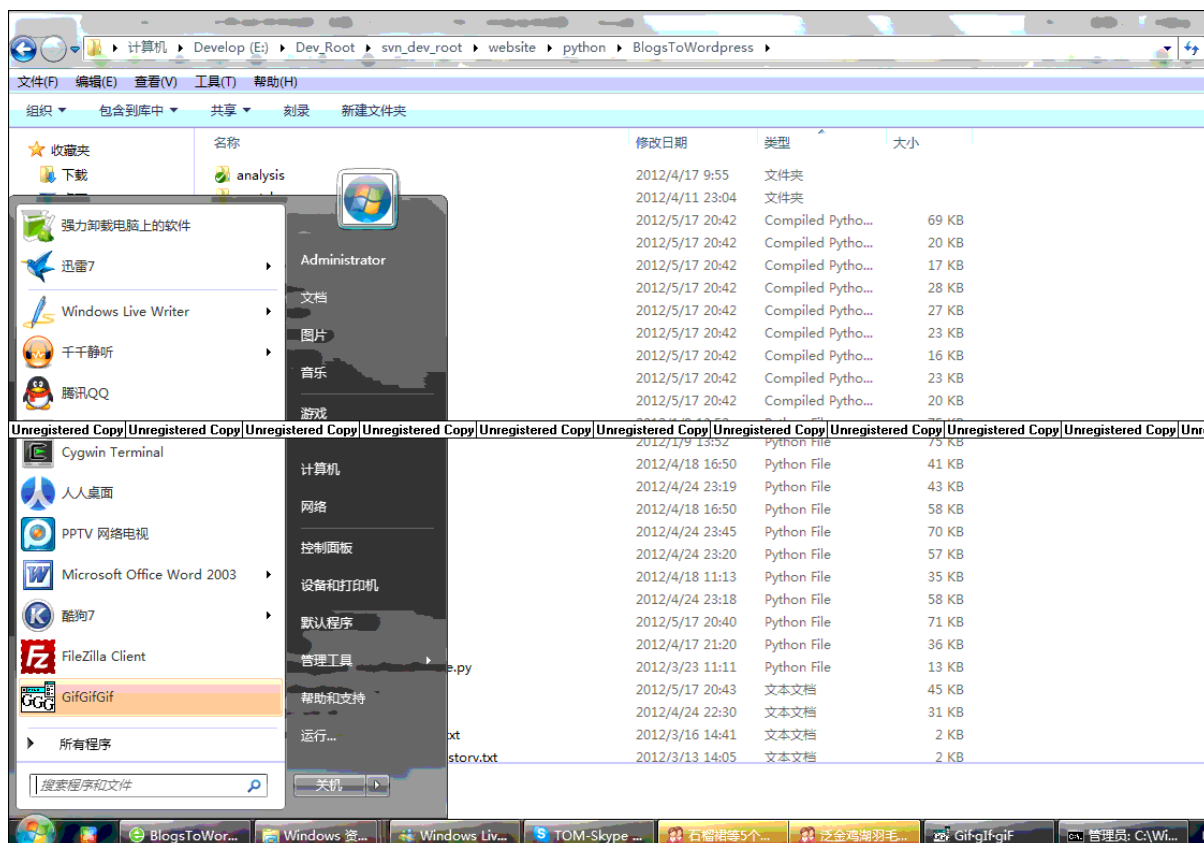


2. 使用Python脚本发布者封装好的，可以独立运行的Python脚本对应的可执行文件
此用法，必须要Python脚本发布者，专门在写完Python脚本的时候，专门去用相应的工具，将Python的文本文件，封装成可以脱离Python运行环境，可以独立运行的可执行文件。说白了，就是一个exe可执行文件，而运行此exe文件，不需要安装Python，即可运行。

4.1.5.1. 如何在Windows下的cmd中运行BlogsToWordpress.py

下面，以Win7为例，通过专门录制的gif动画，来说明，如何在Windows的cmd中，运行我的一个Python脚本：BlogsToWordpress.py

图 4.4. 动画演示如何在Windows的cmd中运行Python脚本BlogsToWordpress.py



提示

上述gif动画演示，只支持HTML在浏览器中的显示。其他格式输出中，比如PDF中，不支持此gif动画。所以下面再用文字解释一下大概流程：[用文字解释如何在Windows的cmd中运行Python脚本BlogsToWordpress.py](#)

如果你是在非HTML格式，比如PDF等，中查看此文，想要看动画的话，猛击[动画演示如何在Windows的cmd中运行Python脚本BlogsToWordpress.py](#)³⁸

用文字简述就是：

用文字解释如何在Windows的cmd中运行Python脚本BlogsToWordpress.py.

1. 打开Windows的命令行环境cmd
开始 ⇒ 在“搜索程序和文件”的位置，输入cmd，win7会自动搜索到cmd，点击其以打开cmd
2. 切换到对应的你的Python脚本所在位置
可以先去拷贝你的Python脚本所在的路径，此处为：

```
E:\Dev_Root\svn_dev_root\website\python\BlogsToWordpress
```

，然后在cmd中输入

```
E:
cd E:\Dev_Root\svn_dev_root\website\python\BlogsToWordpress
```

3. 输入脚本名（及参数），以运行Python脚本
输入相应的python脚本命令：

```
BlogsToWordpress.py -s http://blog.sina.com.cn/lifecoaching
```

然后回车运行该脚本

4.2. 如何在Linux环境下开发Python

介绍如何在Linux环境下运行Python脚本

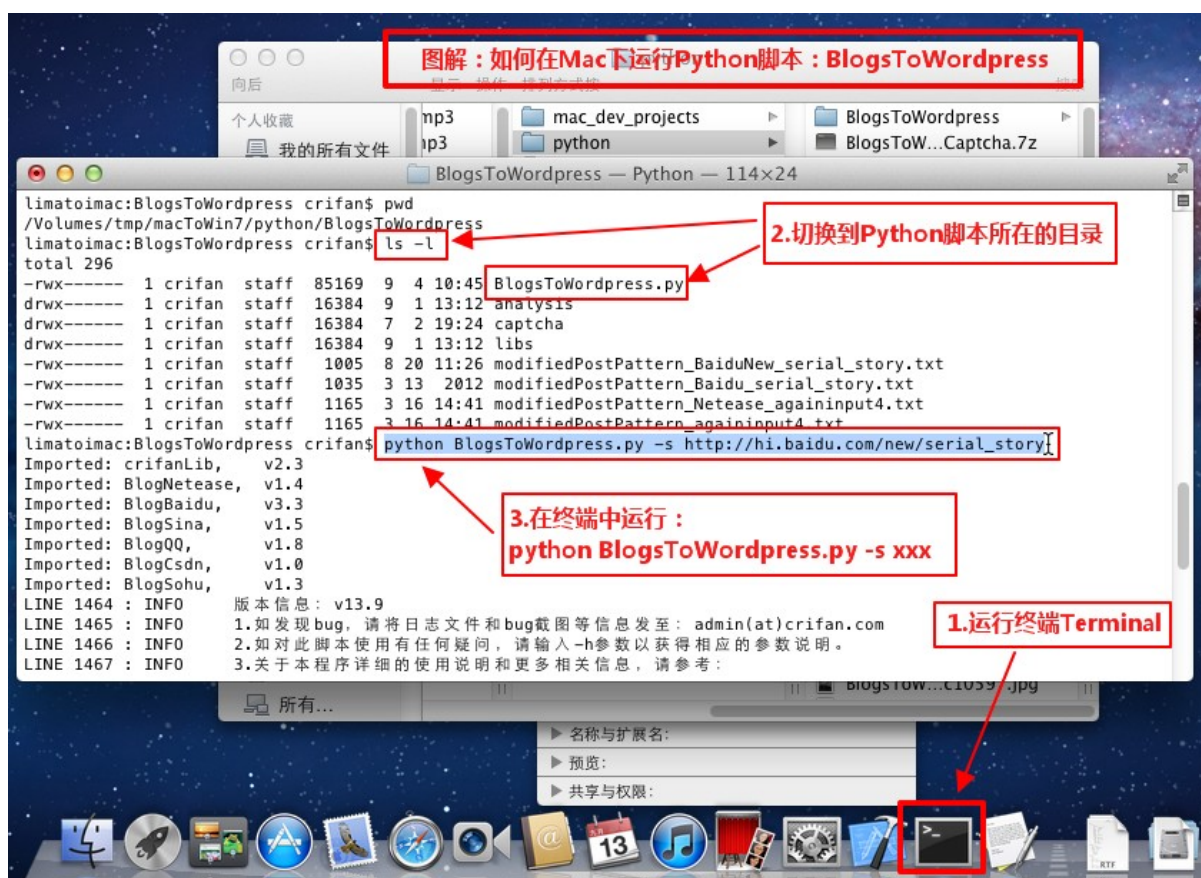
如何进行Python的开发

4.3. 如何在Mac环境下开发Python

Mac下本身已经安装了Python，所以不需要额外安装Python，就可以直接在终端Terminal下去运行Python了。

在Mac下运行Python脚本的方法如下：

图 4.5. 在Mac下的Terminal中运行Python脚本：BlogsToWordpress



第 5 章 Python的基本语法和基础知识

本章主要讲解读者在拥有了自己的开发环境后，需要继续深入学习的Python的基本语法和Python中的一些基本的概念。

此处只介绍，最基本的一些Python的语法，和其他一些基础知识。



相关旧帖

[【整理】Python中的module,library,package之间的区别](#)¹

[【整理】Python中的 __name__ 和 __main__ 含义详解](#)²

5.1. 一张图片入门Python

之前已有别人整理了，[一张图入门Python](#)³，快速了解各种基本的语法。

英文版：

¹ http://www.crifan.com/python_module_vs_library_vs_package

² http://www.crifan.com/python_detailed_explain_about__name__and__main__

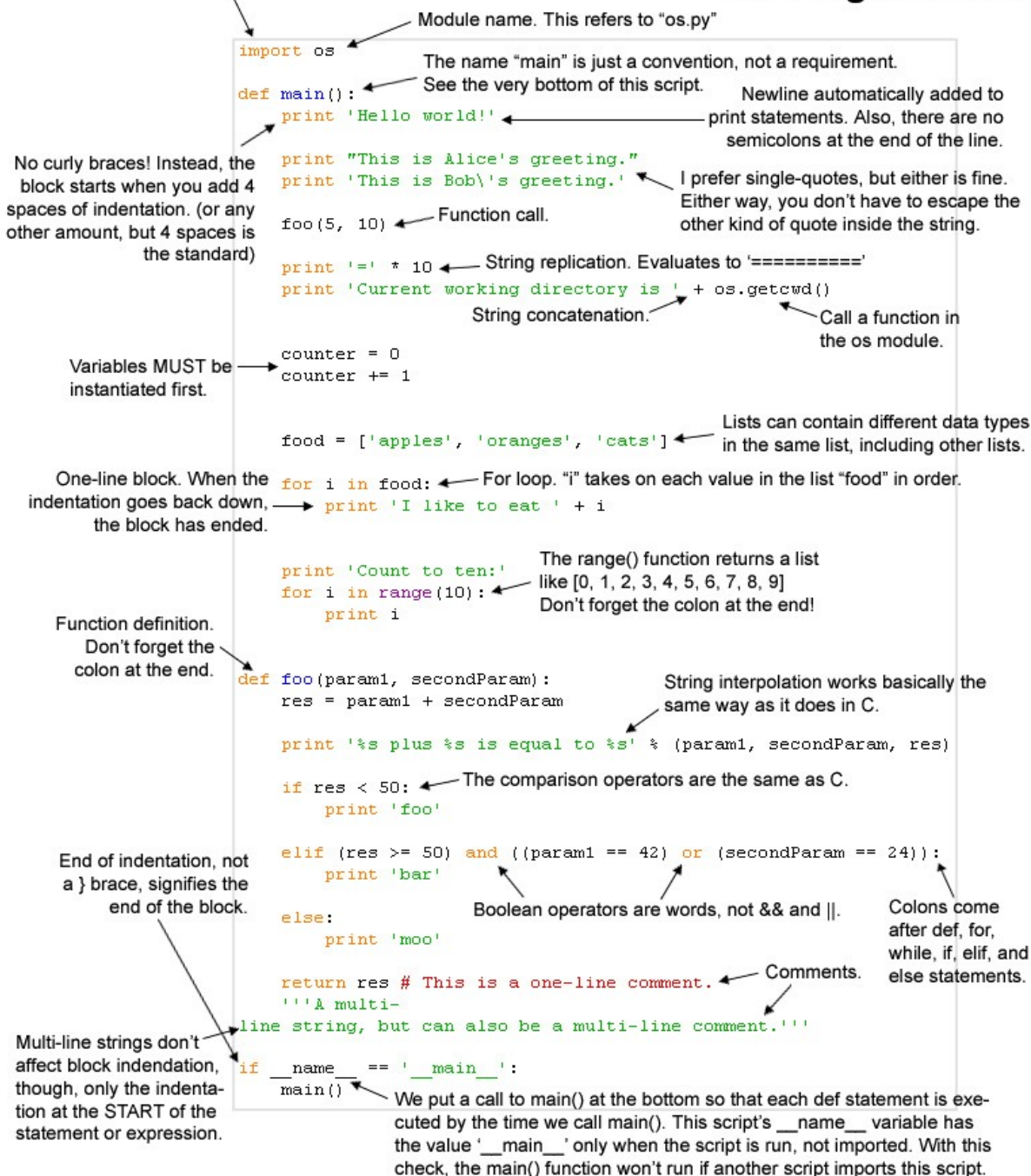
³ <http://www.love67.net/?p=731>

图 5.1. Quick Python Script Explanation

<http://coffeeghost.net>

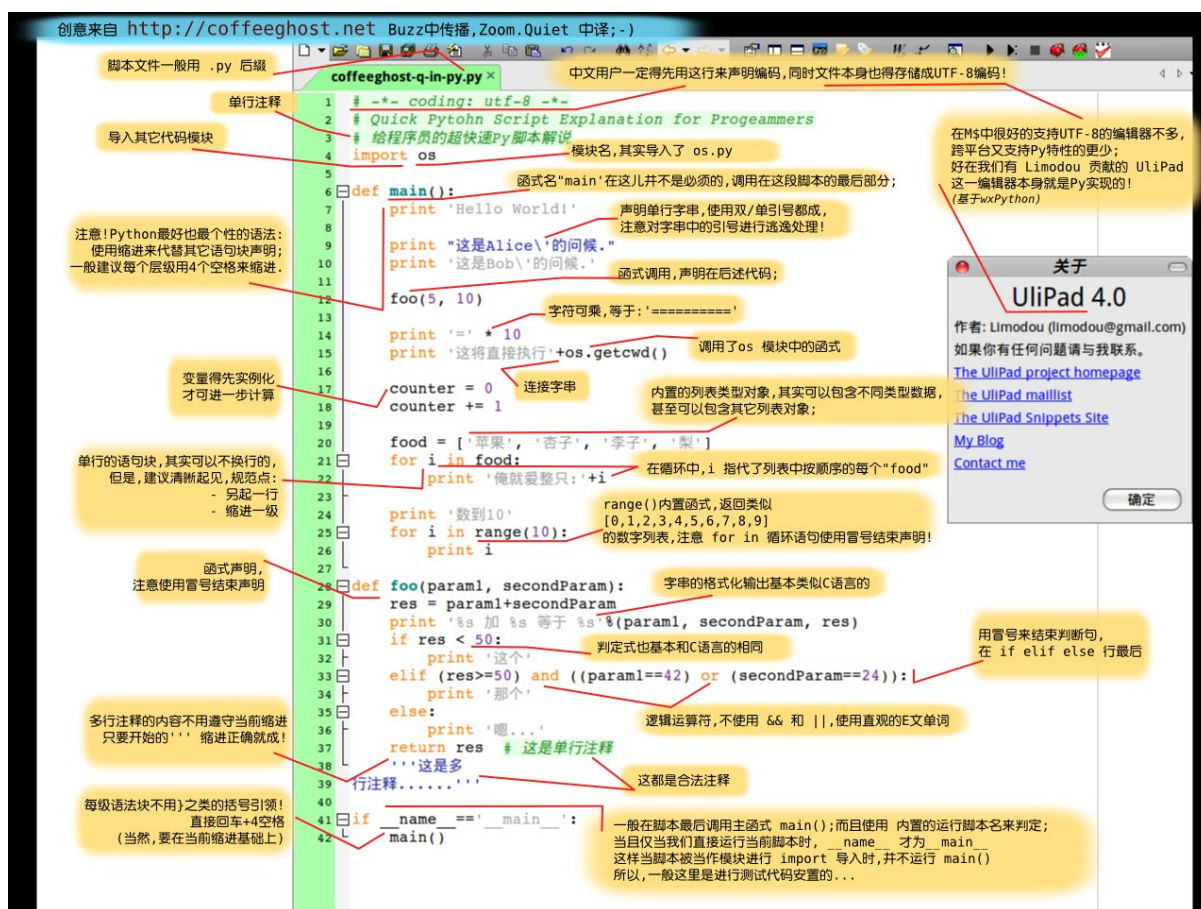
Load other code modules.

Quick Python Script Explanation for Programmers



中文版：

图 5.2. 一张图入门Python中文版



5.2. Python中的2.x版本和3.x版本



相关旧帖

【整理】总结Python2(Python 2.x版本)和Python3 (Python 3.x版本)之间的区别⁴

【整理】关于Python 3.x中, 使用print函数时出现的语法错误 (SyntaxError: invalid syntax) 的问题的原因⁵

在具体介绍, 如何下载Python, 安装Python, 进行Python开发之前, 一个不得不先要解释的话题就是, Python中的版本的问题。

主要包括2.x系列的和3.x系列的, 两者不兼容。

下面详细解释一下。

关于两种版本的对比, 也常写成为:

- Python 2.x vs Python 3.x
- Python 2 vs Python 3
- py2 vs py3

⁴ http://www.crfan.com/summary_the_difference_between_python2_and_python3

⁵ http://www.crfan.com/summary_reason_of_python_3_x_print_syntaxerror_invalid_syntax

抽空参考：

<https://wiki.python.org/moin/Python2orPython3>

去整理一下版本的历史。

5.3. Python文件编码声明

之所以要在介绍Python的语法之前，先去说明这个Python文件的编码声明，

那是因为，更多的Python初学者，对于去参考别人写的Python的文件中的代码的时候，

前几行，就是看到的这部分，

所以，需要先介绍这部分的内容，

使得能在真正开始参考学习Python代码之前，就对于文件编码声明这部分的内容，有个清晰的认识

待整理：

[【整理】关于Python脚本开头两行的：#!/usr/bin/python和# -*- coding: utf-8 -*-的作用 – 指定文件编码类型](#)⁶

5.4. Python中的缩进

此处介绍Python的基本的变量定义等内容之前，需要详细的介绍，关于Python中的缩进

因为，此缩进不仅仅是美观问题，而且还决定了代码的逻辑层次，决定了代码的含义

TODO：添加提示，关于别的语言中，一般来说，代码的缩进与否，都是为了代码更美观，可读性更好，而此处的Python特殊：缩进直接决定了代码的内在逻辑含义。

[【教程】详解Python中代码缩进（Indent）：影响代码的内在逻辑关系和执行结果](#)⁷

5.5. Python中基本变量的声明和定义

Python中基本的变量的声明和定义，初始化，使用

包括：整型，字符串，等等等等

5.5.1. Python中变量的作用域

[【整理】Python中变量的作用域\(variable scope\)](#)⁸

5.5.2. Python中变量与C语言中的变量对比

对于很多人，在开始学习Python之前，往往都是有了一定的C语言的基础

⁶ http://www.crifan.com/python_head_meaning_for_usr_bin_python_coding_utf-8

⁷ http://www.crifan.com/tutorial_python_indent

⁸ http://www.crifan.com/summary_python_variable_effective_scope

而对于C语言，是其他很多语言，比如C，C#等语言的基础，且其他这类语言，其语法，从宏观上来说，

尤其是变量定义和使用，都是相对很类似的

所以，为了使具有了其他语言，尤其是C语言，基础的人，更好的对于Python中的变量有个更深入的了解

此处，专门将Python中的变量，和C语言中的变量，进行对比说明：

TODO：添加对比说明

5.6. Python中的分支结构

写程序，会遇到分支判断

Python也不例外

此处介绍，if/else,switch/case,try/catch等等结构的含义和写法

5.7. Python中的函数

写程序 要有良好的习惯

当代码量相对多了，某些代码属于一个功能集合的时候，就应该去将这部分代码，单独组织到函数里面了

下面介绍Python中的函数的语法和调用

5.8. Python中的面向对象编程

Python中有些语法，是和面向对象编程相关的，比如：

[【整理】Python中：self和__init__ 的含义 + 为何要有self和__init__](http://www.crifan.com/summary_the_meaning_of_self_and__init__in_python_and_why_need_them)⁹

其实，这些部分的内容，如果你是和笔者类似：

只是用到Python的一些去实现一些自己需要的功能，其实可能，用不到这部分的内容

比如我之前用Python实现爬虫的时候，从头到尾，其实一直就没太涉及这部分，关于面向对象方面的内容。

⁹ http://www.crifan.com/summary_the_meaning_of_self_and__init__in_python_and_why_need_them

第 6 章 继续学习Python的思路和方法

本章主要去继续学习Python的时候，如何去学习，尝试教给读者一种思路，以达到授人鱼更授人以渔的目的。

这样，即使以后不看作者的教程，具有了Python的基本知识和学习思路，也就可以靠自己继续学习了。

此处只介绍，在了解了基本的Python的开发知识之后，接下来，想要进一步开发，如何去做，如何学习。

目的在于，掌握了基本的学习方法后，就可以自己去一点点学习更多细节的东西了。

6.1. 如何继续深入学习Python

[【整理】如何学习Python + 如何有效利用Python有关的网络资源 + 如何利用Python自带手册\(Python Manual\)](#)¹

最好有个目的

即，基于一定的兴趣导向，寻找一个合适的目标

比如自己给自己找个合适的任务，要实现一定的功能，最好是能满足自己或他人的实际需求的目标

如此，继续利用相关资源，边学边写，

即一边慢慢搞清楚，需要实现相关的功能，大致思路和要做的事情有哪些，

一边将实际的思路，用真正的Python代码去实现出来

其中遇到相关的Python的基本语法，库的使用，函数的接口等等细节问题

再去参考相关的API手册说明，一点点学习并写出自己的Python代码

举例：

我之前自己就是：

由于需要实现博客搬家，把之前我自己在网易163博客，腾讯的QQ空间，百度的hi空间等等地方的，N多个博客的内容，

都要迁移到我自己的新建的个人网站中

由此：首先是找到了，真实存在的个人需求，需要去解决

然后再去学习和了解，所涉及到的技术，是和网络爬虫有关的

然后再去参考别人写网络爬虫的相关的代码

其中用到了经典的urllib等等库函数，

然后再去参考Python自带手册，一点点学习其详细的语法

最终，从无到有，一点点搞懂了：网络爬虫的实现逻辑，如何用Python实现网络爬虫，Python中和网络处理相关的库函数的使用

由此：最终算是，对于Python语言，有了更进一步的了解，明白并掌握了常见的语法，基本的库函数，基本概念的处理，尤其是字符串等方面的棘手问题

¹ http://www.crifan.com/howto_learn_python_utilize_network_resource_manual

6.2. 如何利用Python相关资源

6.2.1. 如何利用Python自带的手册

TODO : [【整理】如何学习Python + 如何有效利用Python有关的网络资源 + 如何利用Python自带手册\(Python Manual\)](#)²

6.2.2. 如何利用一些在线的Python资源

TODO : [【整理】如何学习Python + 如何有效利用Python有关的网络资源 + 如何利用Python自带手册\(Python Manual\)](#)³

² http://www.crifan.com/howto_learn_python_utilize_network_resource_manual

³ http://www.crifan.com/howto_learn_python_utilize_network_resource_manual

第 7 章 Python常见问题及解答

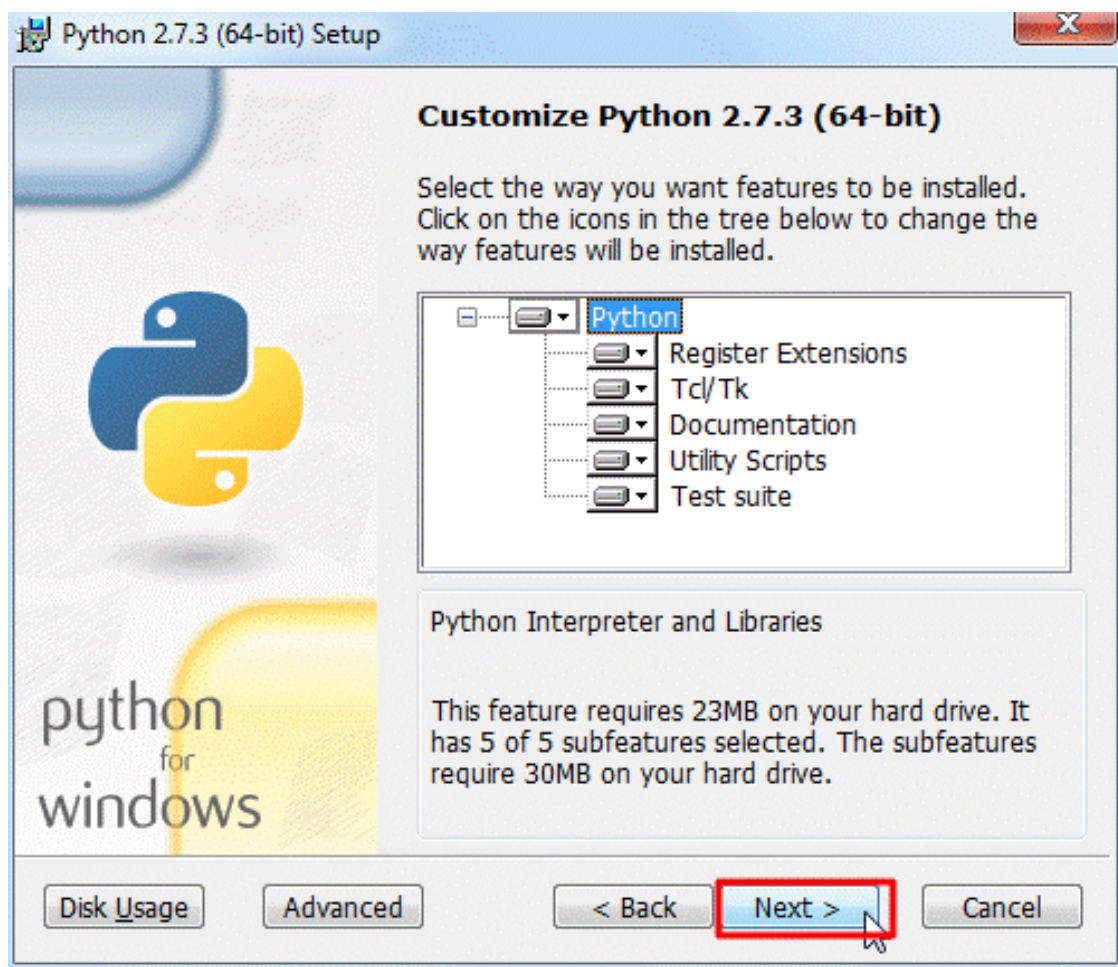
本章主要解释在Python学习期间，常见的一些问题，探讨问题原因，给出解决答案和思路

7.1. 在window的cmd中运行python结果却调用了文本编辑器去打开了，而不是去调用Python解析器去运行python文件

TODO：验证下面的推断是否正确。

估计是，在安装Python时：

图 7.1. 安装Python时选择Register Extensions



没有选择Register Extensions，从而导致之后在cmd中运行py文件，结果被其他默认的文本编辑器去打开，而不是运行Python脚本文件了。

又或者是：

在安装其他文本编辑器，比如Notepad++，然后默认注册了py后缀

导致打开Python的py文件时，默认是调用，默认打开程序，此处为[Notepad++](http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html)¹去打开。

¹ http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html

第 8 章 Python相关资源

此处整理一些Python的学习资料。

TODO：整理更多的有价值的参考资料。

此处记录一些不错的Python相关的资料，供需要时参考：

- [简明Python教程](#)¹
Swaroop, C. H. 著, >沈洁元 译
特点：内容很全。
- [深入 Python 3](#)²
- 关于python的中文文档
这里：<http://www.elias.cn/Python/HomePage>介绍了很多的资源。其中就有[python教程的中文版](#)³
- 在线实时演示Python代码的运行
[Online Python Tutor - Learn programming by visualizing code execution](#)⁴
- Python的示例程序
[SimplePrograms - Python Wiki](#)⁵
- 各种Python的示例代码
[Python examples \(example source code\) Organized by topic](#)⁶
- Python风格的详尽解释和举例
[Code Like a Pythonista: Idiomatic Python](#)⁷
- 如何写自己的库函数
<http://postgetter-app.googlecode.com/hg-history/04cc032892e8a81a46eeb15ec7814fc3b39ed6ab/>
[PostGetter.py](#)值得学习学习。关于自己写类，库函数时，如何写，如何处理logger和异常等等方面，值得学习。
- Python资料大全
<http://simple-is-better.com/sites/>收集了N多关于的Python的网站、教程、图书、框架、应用等等内容。
- 关于有介绍Python的IDE
<https://wiki.python.org/moin/IntegratedDevelopmentEnvironments>
- MicroPython
之前看到一个好玩的，另类版本的Python：
[MicroPython](#)⁸
[Micro Python: Python for microcontrollers](#)⁹

¹ http://woodpecker.org.cn/abyteofpython_cn/chinese/

² <http://sebug.net/paper/books/dive-into-python3/>

³ http://wiki.woodpecker.org.cn/moin/March_Liu/PyTutorial

⁴ <http://pythontutor.com/>

⁵ <https://wiki.python.org/moin/SimplePrograms>

⁶ <http://www.java2s.com/Code/Python/CatalogPython.htm>

⁷ <http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html>

⁸ <http://micropython.org/>

⁹ <http://www.kickstarter.com/projects/214379695/micro-python-python-for-microcontrollers>

参考书目

- [1] [Python基础篇](#)¹
- [2] [python编辑器对比和推荐](#)²
- [3] [Python - 维基百科，自由的百科全书](#)³
- [4] [【整理】【多图详解】如何在Windows下开发Python：在cmd下运行Python脚本，如何使用Python Shell（command line模式和GUI模式），如何使用Python IDE](#)⁴
- [5] [【整理】计算机语言基础知识介绍](#)⁵
- [6] [【整理】Python语言简介](#)⁶
- [7] [【整理】总结Python2\(Python 2.x版本\)和Python3\(Python 3.x版本\)之间的区别](#)⁷
- [8] [【crifan推荐】轻量级文本编辑器，Notepad最佳替代品：Notepad++](#)⁸
- [9] [Windows的命令行工具: cmd](#)⁹
- [10] [【整理】Python中的图形库](#)¹⁰
- [11] [【已解决】在用google搜索出来的链接无法打开的情况下，如何找到该链接的真实地址](#)¹¹
- [12] [【整理】什么是IDE](#)¹²
- [13] [【整理】各种Python的IDE\(集成开发环境\)的总结和对比](#)¹³
- [14] [【记录】折腾Python中的Tkinter](#)¹⁴
- [15] [【记录】使用Python的IDE：PyScripter](#)¹⁵
- [16] [【整理】Python中的 name 和 main 含义详解](#)¹⁶
- [17] [Eclipse+PyDev](#)¹⁷

¹ http://www.tsnc.edu.cn/default/tsnc_wgrj/doc/python/basic.htm

² <http://blog.csdn.net/cserchen/article/details/7036435>

³ <http://zh.wikipedia.org/wiki/Python>

⁴ http://www.crifan.com/how_to_do_python_development_under_windows_environment

⁵ http://www.crifan.com/computer_language_basic_knowledge_introduction

⁶ http://www.crifan.com/simple_intro_what_is_python_and_how_to_run_python_script

⁷ http://www.crifan.com/summary_the_difference_between_python2_and_python3

⁸ http://www.crifan.com/files/doc/docbook/rec_soft_npp/release/html/rec_soft_npp.html

⁹ http://www.crifan.com/files/doc/docbook/soft_dev_basic/release/html/soft_dev_basic.html#win_cmd

¹⁰ http://www.crifan.com/summary_python_graphics_gui_libs_packages/

¹¹ http://www.crifan.com/find_real_link_from_google_link_when_failed_open_via_google/

¹² http://www.crifan.com/what_is_ide

¹³ http://www.crifan.com/summary_common_python_ide_pyscripter_ulipad_eclipse_pydev_eric

¹⁴ http://www.crifan.com/try_python_tkinter_module/

¹⁵ http://www.crifan.com/try_with_python_ide_pyscripter/

¹⁶ http://www.crifan.com/python_detailed_explain_about__name__and__main__/

¹⁷ http://www.crifan.com/try_with_python_ide_eclipse_pydev