**Python学习和开发环境的建立**

Python是一种面向对象、直译式的计算机程序设计语言，也是一种功能强大的通用型语言，已经具有近二十年的发展历史，成熟且稳定。它包含了一组完善而且容易理解的标准库，能够轻松完成很多常见的任务。Python的语法非常简捷和清晰，与其它计算机程序设计语言最大的不同在于，它采用缩进来定义语句块。

Python的官方介绍是：Python是一种简单易学，功能强大的编程语言，它有高效率的高层数据结构，能简单而有效地实现面向对象编程。Python简洁的语法和对动态输入的支持，再加上解释性语言的本质，使得它在大多数平台上的很多领域都是一个理想的脚本语言，特别适用于快速的应用程序开发。

Python支持命令式编程、面向对象程序设计、函数式编程、面向切面编程、泛型编程多种编程范式。与Scheme、Ruby、Perl、Tcl等动态语言一样，Python具备垃圾自动回收功能，能够自动管理内存使用。Python经常被当作脚本语言用于处理系统管理任务和Web编程，然而它也非常适合完成各种高阶任务。 Python虚拟机本身几乎可以在所有的操作系统中运行。使用一些诸如py2exe、PyPy、PyInstaller之类的工具可以将Python源代码转换成可以脱离Python解释器执行的程序。

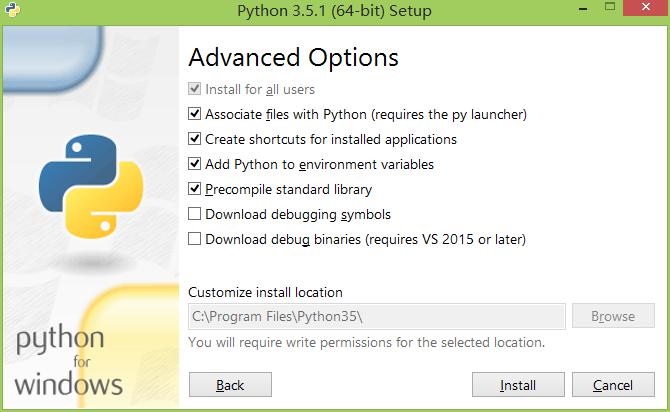
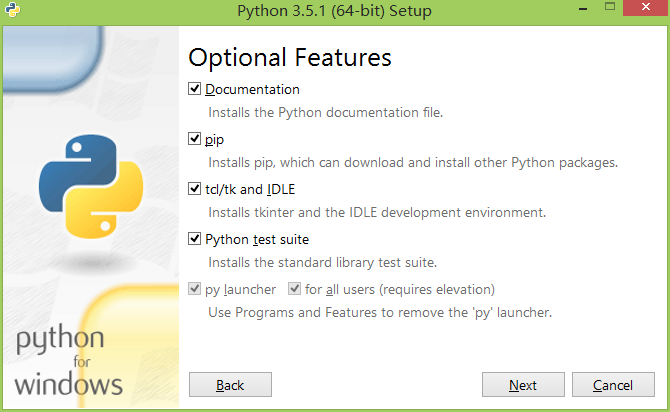
Python的主要参考实现是CPython，它是一个由社区驱动的自由软件。目前由Python软件基金会管理。基于这种语言的相关技术正在飞速的发展，用户数量急剧扩大，相关的资源非常多。

Python解释器、标准库及扩展包构成Python的基本开发环境，满足学习和软件开发的要求。Python解释器及其标准库可从Python官网下载；其他扩展包可用pip命令等方法安装。

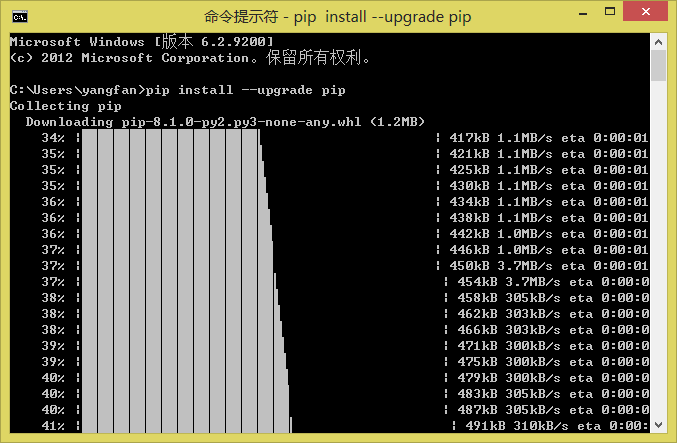
大型软件开发宜采用集成开发工具，常用的有: Eclipse，VS2015等，其中，Eclipse加PyDev插件的方式， 为较多Python程序开发人员使用。此外，还有使用编辑器(如Atom)构建开发环境的方法。此文介绍Python安装配置及Eclipse加PyDev插件开发工具的安装。

一、Python基本开发环境的建立

从官网：<http://www.python.org> 下载和操作系统（32/64位）对应版本的Python3软件。下面以Python3.4.3版本安装为例做简单说明

 Windows： 建议使用定制方式安装: 1) 加环境变量PATH; 2) 自定义安装目录： C:\python35

然后，命令行下更新pip到新版本：

python -m pip install -U pip

二、Python扩展包安装

pip命令是软件包的基本安装方式，不过，有些软件包使用pip安装时，需要本地编译，编译条件如果不满足安装就会失败。

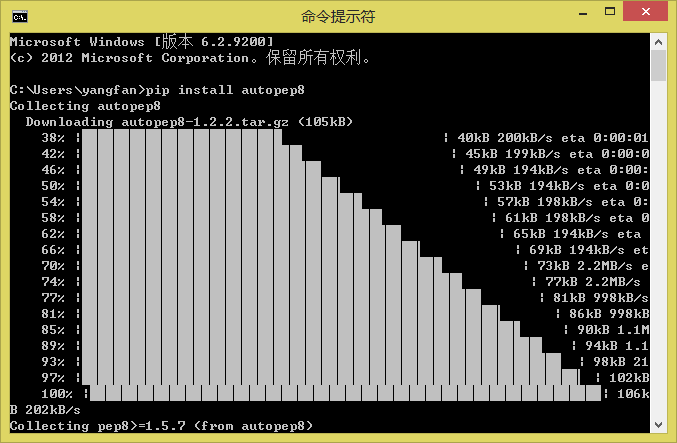
这时可使用： 加州大学欧文分校提供的Windows32/64位Python扩展包编译版，

<http://www.lfd.uci.edu/~gohlke/pythonlibs/>

加州大学欧文分校的编译版包，内容丰富、更新及时、安装方便。

2.1 编码规范

pip install autopep8



pip install pylint（方法如上）

2.2 科学计算软件包

**2.2.1 SciPy**

Scipy官网：<http://www.scipy.org/>,提供有全面的科学计算软件包信息。基本应用安装以下3个即可：

numpy

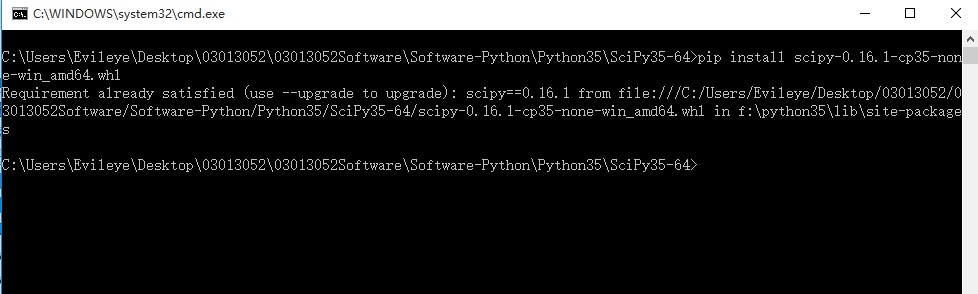
scipy

matplotlib

Window环境下Scipy 软件包的 安装，建议从加州大学欧文分校下载编译好的\*.whl，然后，用pip逐个本地安装：

>pip install \*.whl

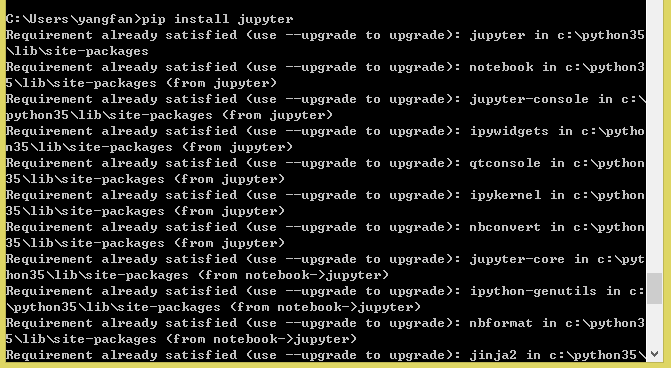
示例安装scipy：找到下载whl文件所在位置，然后shift+鼠标右键在此处打开命令窗口，键入>pip install scipy-0.16.1-cp35-none-win\_amd64.whl



2.2.2 交互计算 Jupiter

**1） 安装Jupiter ：**

>pip install jupyter 在线安装



**2） 安装Python语言内核**

>pip install ipython ，支持Python语言（方法如上图）

**3） 安装依赖包**

>pip install pyreadline

>pip install sympy （方法如上图）

**4）支持显示数学符号、公式，安装MathJax：**

命令行下，

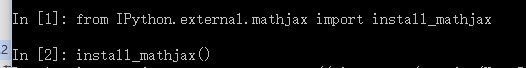
>IPython

打开IPythonshell，然后，在In[1]:后键入：

from IPython.external.mathjax import install\_mathjax

在In[2]:后键入代码：

install\_mathjax()



**5） 运行notebook:**

在iPython notebook文件所在目录下，打开命令行窗口：

>jupyter notebook

2.2.3 IF97 for Python

Windows 32/64位版: 从

<https://github.com/Py03013052/SEUIF97>

下载：SEUIF97.dll和 seuif97.py，然后：

1) SEUIF97.dll拷贝到 c:\windows\system

2) seuif97.py拷贝到 c:\python35\Lib

三、基于Eclipse的Python开发环境

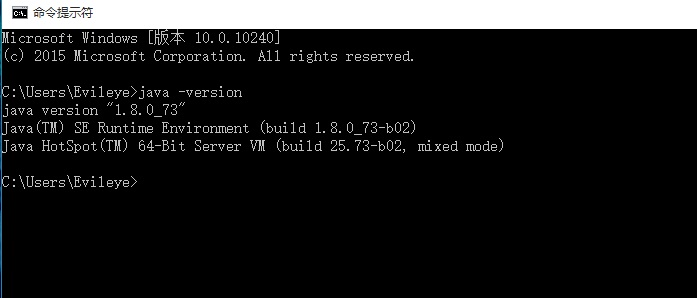
3.1 安装Eclipse IDE

**3.1.1安装JavaSDK**

Eclipse IDE是使用Java开发的，电脑中需要预先安装好Java JRE/JDK软件包，因此，安装前，在命名行下，输入：

>java –version

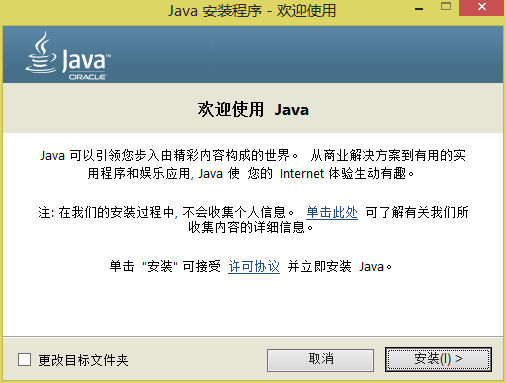
检查是否已经安装了Java软件包。 如果电脑中已经安装了Java ,会显示有关版本。



如果没有安装java 或者版本不能满足Eclipse IDE最低要求，需要从Oracle下载Java包。

Java JDK的Oracle官方下载地址如下：

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>



下载时需要注意操作系统32/64位，选择相应版本下载（\*是Java版本号）：

Windows 32位：\*-windows-i586.exe

Windows 64位：\*-windows-i64.exe.

运行下载Oracle的Java安装包（安装时注意配置环境变量），即可安装好Java环境。

**3.1.2 Eclipse IDE**

　　Eclipse IDE是插件型开发环境，有很多版本可以下载。本课程主要使用Python语言，会涉及C/C++开发，所以，建议下载Eclipse CDT(Eclipse IDE　for　C/C++)版.

Eclipse CDT官方下载地址：

http://www.eclipse.org/downloads/



根据操作系统32/63位，下载相应的版本，然后将下载的Eclipse CDT解压到指定目录下，运行解压目录下的：eclipse.exe即可。

如果使用Windows7 以上版本操作系统，建议将运行eclipse.exe，固定到任务栏。（在eclipse.exe文件名上，点鼠标右键即可）





以后，从任务栏启动Eclipse方便很多（同样也可以将IDLE固定到任务栏）。

**3.2 安装Python开发插件PyDev**

使用Eclipse IDE作为Python开发环境，需要：1） 安装PyDev插件；2）配置解释器为Python3.4；

**3.2.1 安装PyDev插件**

通过Windows->Eclipse Marketplaces进入市场，输入Pydev，找到Pydev安装/更新项目，在线安装即可：



在线安装过程简单，但是，安装速度受网络环境影响。

如果很慢，可从pydev官网（<http://www.pydev.org/>）提供的下载地址：

http://sourceforge.net/projects/pydev/files/

下载插件包，然后，通过：

Help->Install->New Software->Add->Archive

离线安装.



**3.2.2 配置 Python解释器**

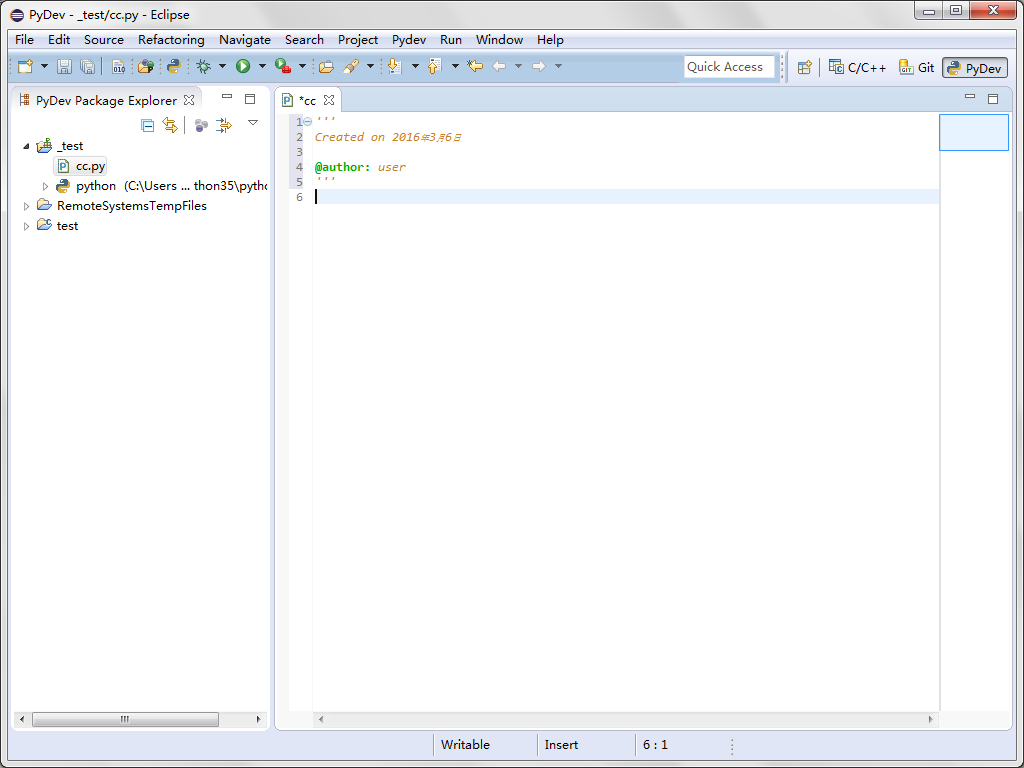
安装好后，重新启动。通过：

Windows->Preference->Pydev->Interperters->Python Interperter点其中的：Advanced Auto-config配置开发使用的 Python解释器版本:



配置好后切换到Python场景，就可以进行Python开发了。

**四、配置 PyDev提高效率**

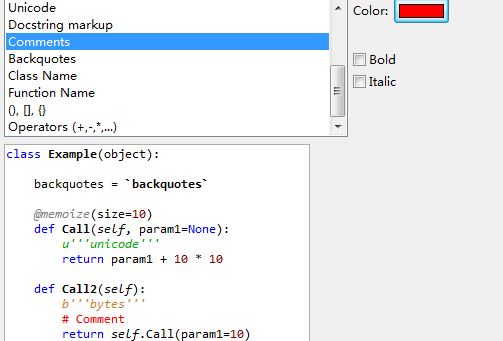


4.1 显示源码行号

右键源码的左边缘，选中Show Line Number（已经默认选中）。

4.2 修改注释颜色提高可读行

从Window → Preferences→PyDev→Editor,进入配置界面：



**五、 基于PyDev的语言规范静态检查**

PyDev中集成了PEP8,AutoPEP8 和Pylint代码检查功能，这些功能默认状态都是关闭的。

程序开发过程中，要有规范意识，但不可能有很高的规范性，过分注意规范会影响开发进程。这时如果一直开启代码规范检查，经常提示不规范，会对开发形成负面影响，所以，默认关闭是合适的。在程序开发一个阶段结果出来时，进行规范性检查更好。

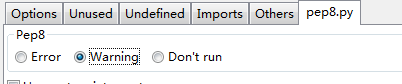
5.1 PEP8检查和修改

1）启动 pep8 检查：

Window > Preferences

PyDev > Editor > Code Analysis > pep8.py

选择Errors/Warnings其中之一..



2）启动autopep8自动修改：

点Windows -> Preferences ->输入 'autopep8' 作为搜索串.

选择（Check）: Use autopep8.py for code formatting?

|  |
| --- |
|  |

在Python源码窗口，按 CTRL-SHIFT-F 就可以自动修改代码

5.2 使用Pylint

PyDev 默认不开启Pylint。通过

     Window -> preferences -> Pydev -> Pylint,选中"Use pylint?",

小结：由于在软件的安装时还不是很了解作业的要求，所以有的图并没有截取下来，向同学加以借鉴。这次只是粗略地了解了python的开发环境和一些重要的插件。还是相当生疏，希望以后能够更加深入地了解学习。