徐静文

03013310

**Python开发环境的建立指导手册**

目录

[**Python学习和开发环境的建立** 2](#_Toc445895947)

[**一 Python基本开发环境建立** 2](#_Toc445895948)

[**二 Python扩展包安装Python基本开发环境建立** 3](#_Toc445895949)

[**2.1 编码规范** 3](#_Toc445895950)

[**2.1.1 autopep8** 3](#_Toc445895951)

[**2.1.2 pylint** 4](#_Toc445895952)

[**2.2 科学计算软件包** 4](#_Toc445895953)

[**2.2.1 SciPy** 4](#_Toc445895954)

[**2.2.2 交互计算 Jupiter** 5](#_Toc445895955)

[**2.2.3 IF97 for Python** 8](#_Toc445895956)

[**三 基于Eclipse的Python开发环境** 8](#_Toc445895957)

[**3.1 安装Eclipse IDE** 9](#_Toc445895958)

[**3.1.1安装JavaSDK** 9](#_Toc445895959)

[**3.1.2 Eclipse IDE** 10](#_Toc445895960)

[**3.2 安装Python开发插件PyDev** 10](#_Toc445895961)

[**3.2.1 安装PyDev插件** 11](#_Toc445895962)

[**3.2.2 配置 Python解释器** 12](#_Toc445895963)

[**3.3 联机指南** 13](#_Toc445895964)

[**四 配置 PyDev提高效率** 13](#_Toc445895965)

[**4.1 显示源码行号** 13](#_Toc445895966)

[**4.2 修改注释颜色提高可读行** 13](#_Toc445895967)

[**4.3 配置任务标签** 14](#_Toc445895968)

[**五 基于PyDev的语言规范静态检查** 15](#_Toc445895969)

[**5.1 PEP8检查和修改** 15](#_Toc445895970)

[**5.1.1 启动 pep8 检查** 15](#_Toc445895971)

[**5.1.2 启动autopep8自动修改** 16](#_Toc445895972)

[**5.2 使用Pylint** 16](#_Toc445895973)

[**小结** 17](#_Toc445895974)

[**参考文献** 18](#_Toc445895975)

**Python学习和开发环境的建立**

Python 是一种面向对象、解释性计算机程序设计语言，于1989年底发明，第一个公开发行版发行于1991年，Python 源代码同样遵循 GPL(GNU General Public License)协议。Python语法简洁而清晰，具有丰富和强大的类库。它常被昵称为胶水语言，能够把用其他语言制作的各种模块(尤其是C/C++)很轻松地联结在一起。常见的一种应用情形是，使用Python快速生成程序的原型(有时甚至是程序的最终界面)，然后对其中有特别要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，性能要求特别高，就可以用C/C++重写，而后封装为Python可以调用的扩展类库。需要注意的是在您使用扩展类库时可能需要考虑平台问题，某些可能不提供跨平台的实现。

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。

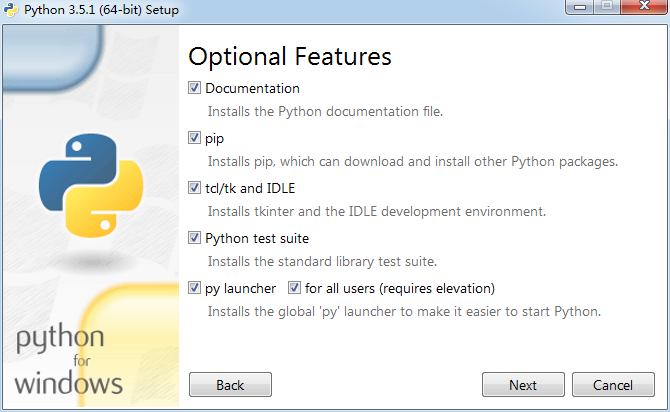
大型软件开发宜采用集成开发工具，常用的有: Eclipse，VS2015等，其中，Eclipse加PyDev插件的方式， 为较多Python程序开发人员使用。此外，还有使用编辑器(如Atom)构建开发环境的方法。

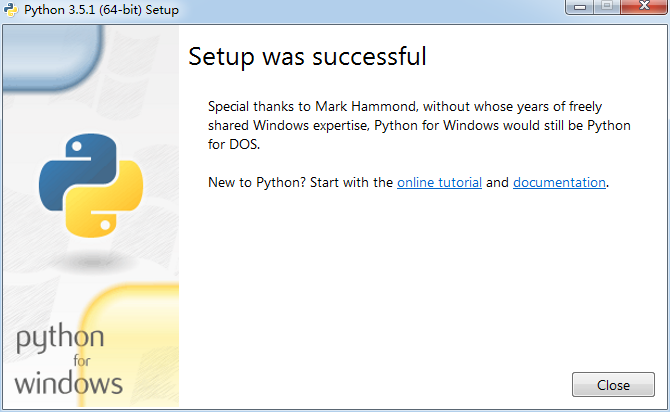
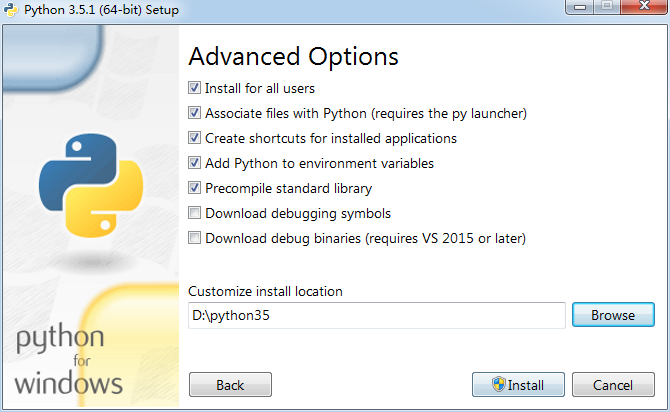
**一 Python基本开发环境建立**

用03013052中分享的安装文件安装Python3.5

Windows系统安装： 1) 加环境变量PATH;

2) 自定义安装目录： D: \python35

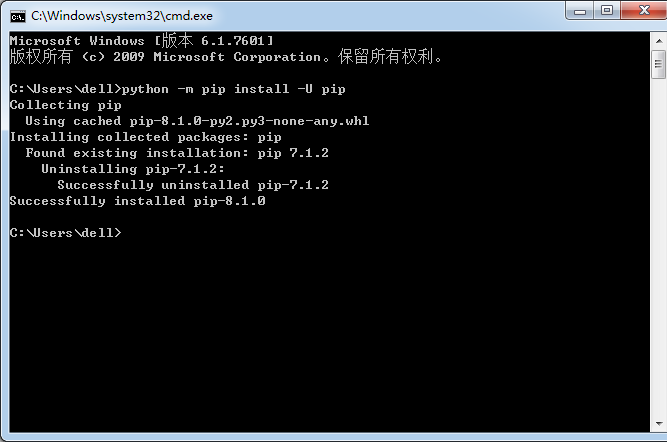




安装成功后，命令行（cmd）下更新pip到新版本：

若输入>pip install --upgrade pip无法完成更新，应输入

>python -m pip install -U pip



**二 Python扩展包安装Python基本开发环境建立**

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。

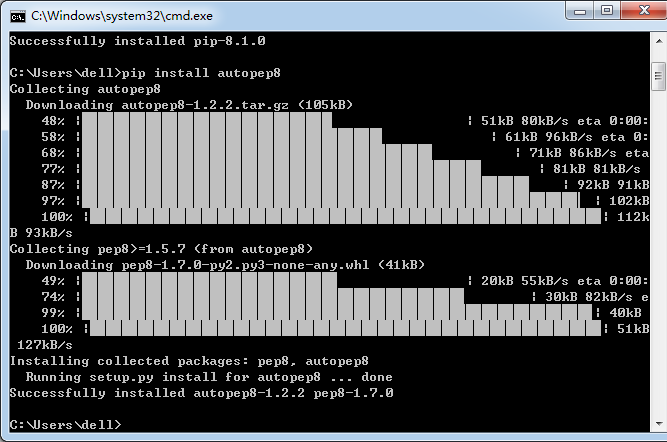
这时可使用： 加州大学欧文分校提供的Windows32/64位Python扩展包编译版，<http://www.lfd.uci.edu/~gohlke/pythonlibs/> 加州大学欧文分校的编译版包，内容丰富、更新及时、安装方便。

**2.1 编码规范**

**2.1.1 autopep8**

Autopep8是一个将Python代码自动排版为PEP8风格的小工具。它使用pep8工具来决定代码中的哪部分需要被排版。

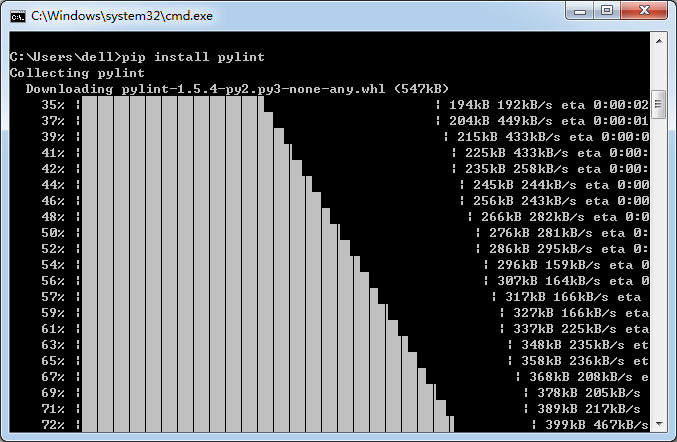
输入命令：pip install autopep8

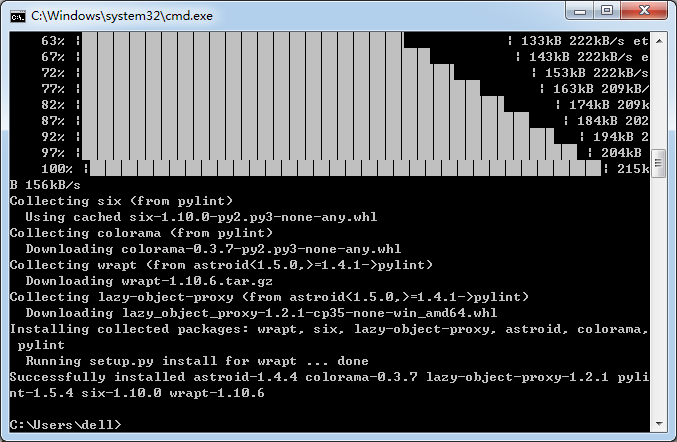


**2.1.2 pylint**

pylint 是一个python代码风格的检查工具, 它依据的标准是 Guido 的 [PEP8](http://www.python.org/dev/peps/pep-0008/). pylint 类似于 [PyChecker](http://pychecker.sourceforge.net/), 但提供了更多的功能, 如检查代码行的长度, 检查变量命名是否符合编码规范, 或检查声明的接口是否被真正的实现，pylint 的最大优势在于其高度的可配置化和可定制化，你可以很容易地写一个小插件添加个人功能。

输入命令：pip install pylint





**2.2 科学计算软件包**

**2.2.1 SciPy**

SciPy是另一种使用NumPy来做高等数学、信号处理、优化、统计和许多其它科学任务的语言扩展。

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

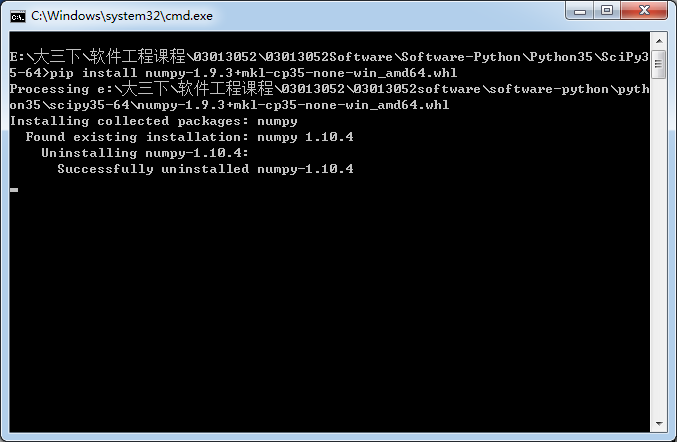
scipy

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

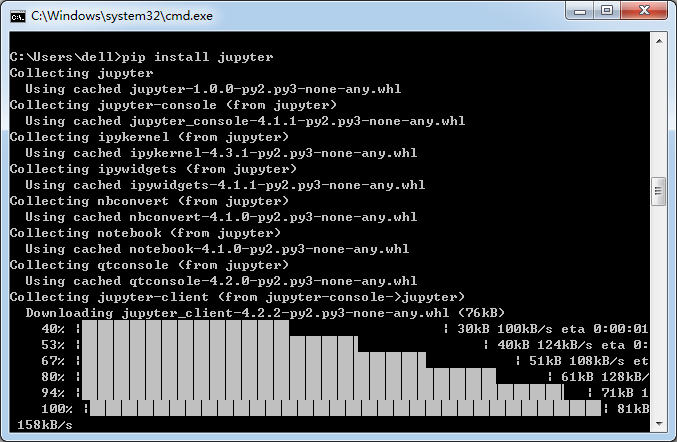
科学计算软件包有两种安装方法，除按上述步骤安装外，还可以直接点击老师给的软件包中window批处理文件即可直接安装

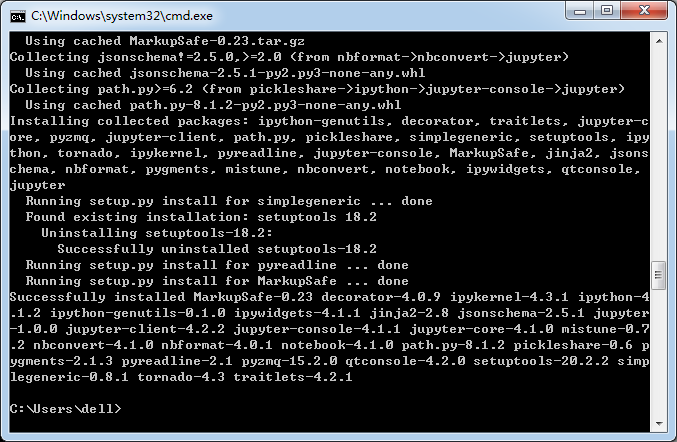


**2.2.2 交互计算 Jupiter**

1） 安装Jupiter ：

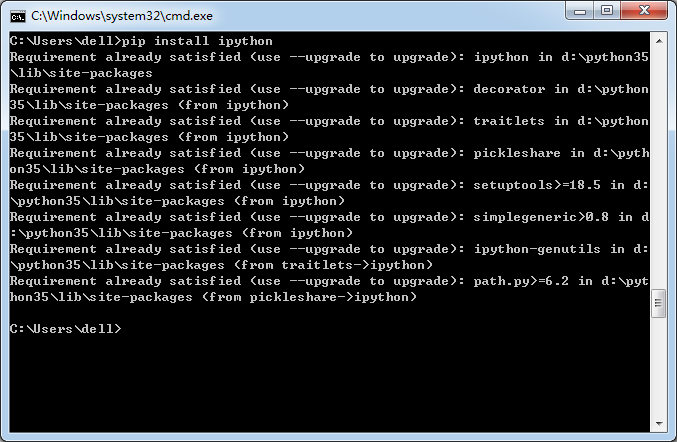
>pip install jupyter 在线安装





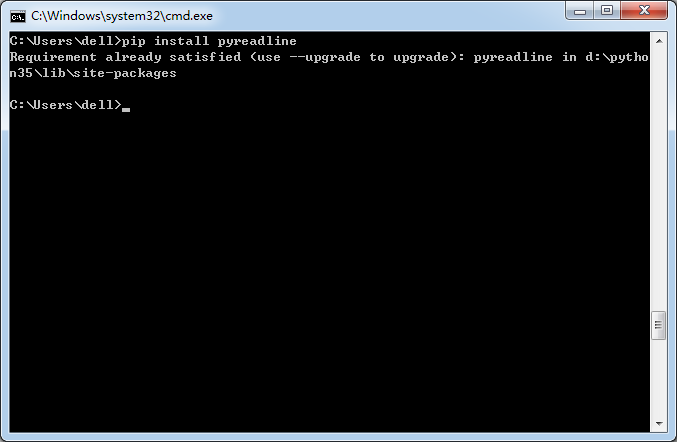
2） 安装Python语言内核

>pip install ipython ，支持Python语言

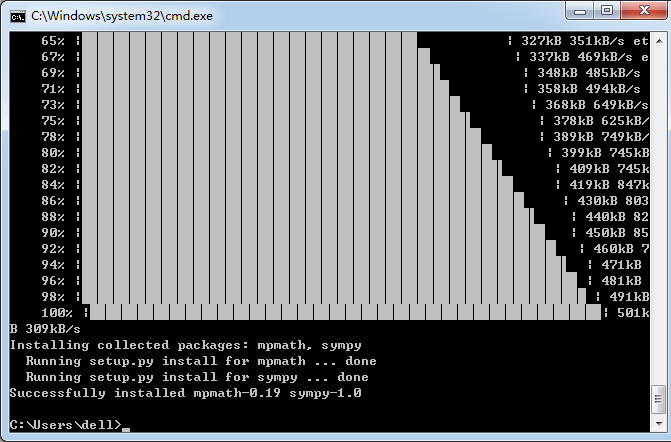


3） 安装依赖包

>pip install pyreadline



>pip install sympy



4）支持显示数学符号、公式，安装MathJax：

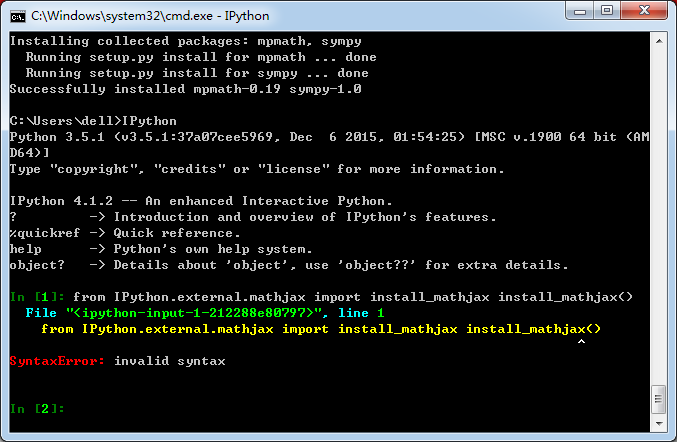
命令行下，

>IPython

打开一个IPython的shell，然后，在其中键入如下代码：

from IPython.external.mathjax import install\_mathjax

install\_mathjax()

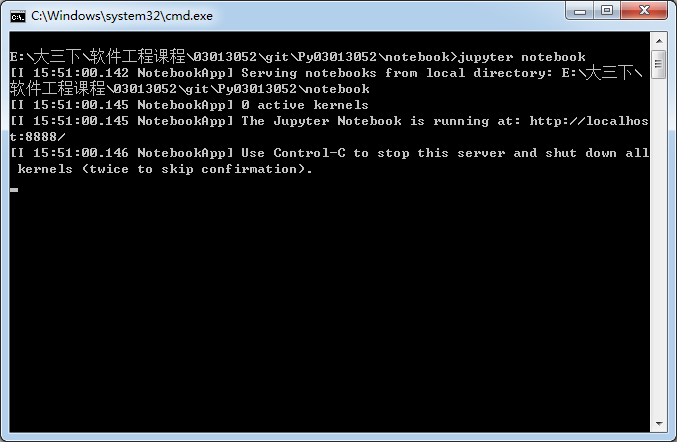


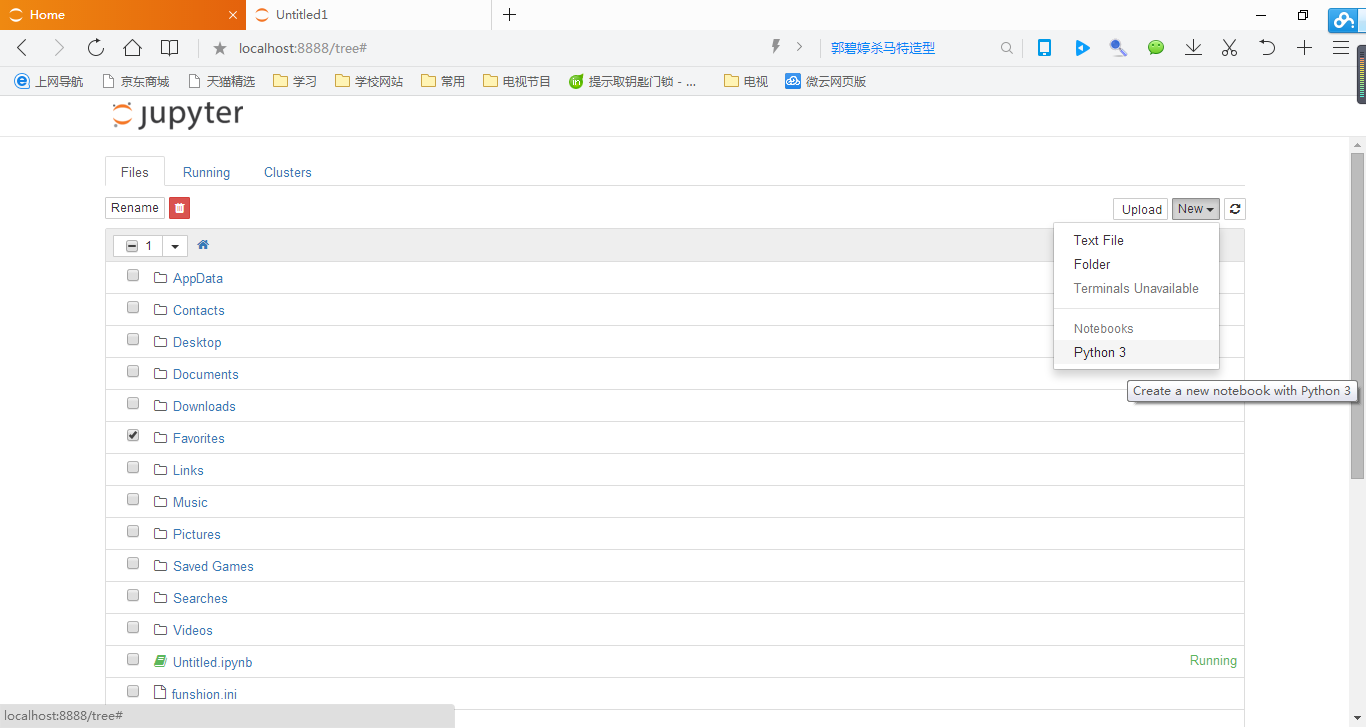
因为安装的Jupyter 是新版，已经默认安装mathjax，故可省略此步骤。

5） 运行notebook:

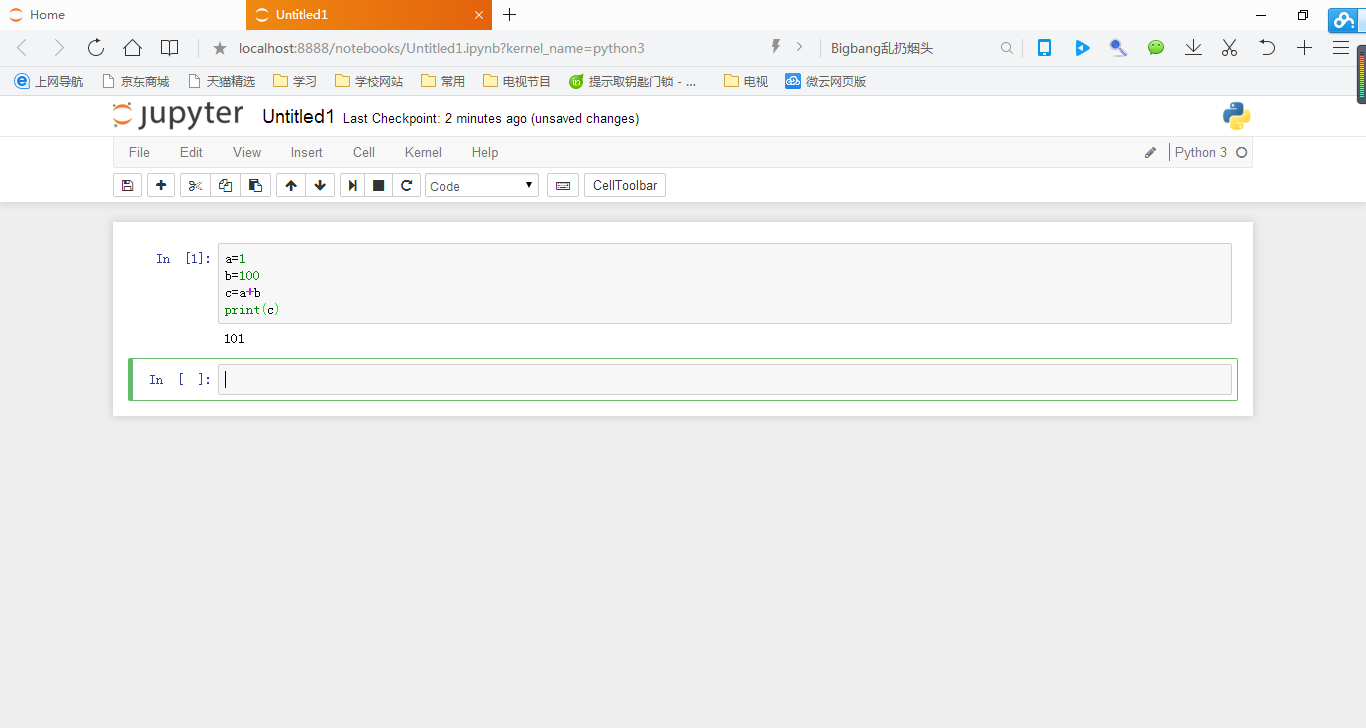
在iPython notebook文件所在目录下，打开命令行窗口：

>jupyter notebook





在此页面可新建一个Python文件，然后可在新建文件中进行简单运算。

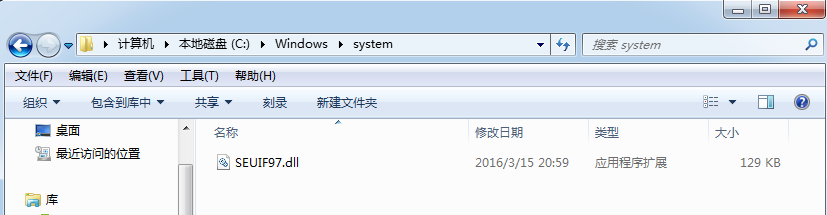


**2.2.3 IF97 for Python**

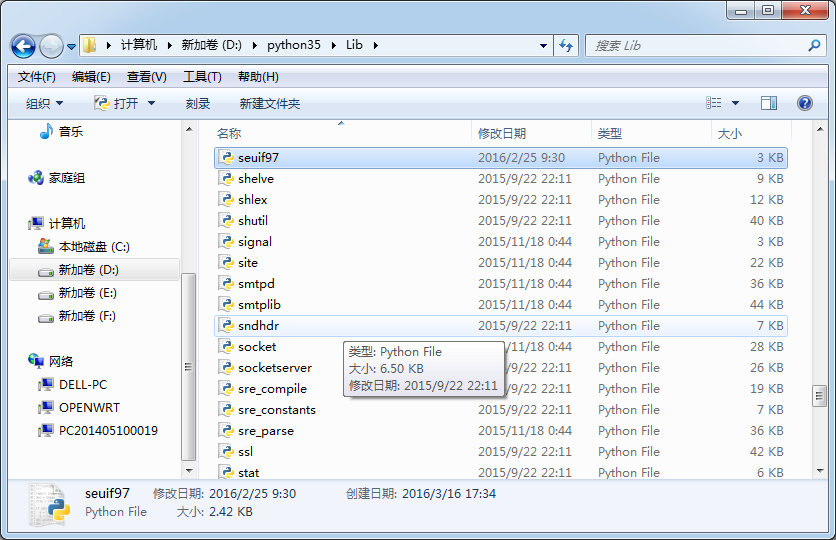
Windows 32/64位版: 从 <https://github.com/Py03013052/SEUIF97>

下载：SEUIF97.dll和 seuif97.py，然后：

1）SEUIF97.dll拷贝到 c:\windows\system



2) seuif97.py拷贝到 D:\python35\Lib



**三 基于Eclipse的Python开发环境**

Eclipse 是一个开放[源代码](http://baike.so.com/doc/5395725-7589480.html" \t "_blank)的、基于[Java](http://baike.so.com/doc/2886868-3046592.html)的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse 附带了一个标准的插件集，包括Java[开发工具](http://baike.so.com/doc/6917134-7139022.html)(Java Development Kit，[JDK](http://baike.so.com/doc/2360646-2496311.html" \t "_blank))。

虽然大多数用户很乐于将 Eclipse 当作 Java [集成开发环境](http://baike.so.com/doc/5602996-5815603.html)(IDE)来使用，但 Eclipse 的目标却不仅限于此。Eclipse 还包括插件开发环境(Plug-in Development Environment，PDE)，这个组件主要针对希望扩展 Eclipse 的软件开发人员，因为它允许他们构建与 Eclipse 环境无缝集成的工具。由于 Eclipse 中的每样东西都是插件，对于给 Eclipse 提供插件，以及给用户提供一致和统一的集成开发环境而言，所有工具开发人员都具有同等的发挥场所。

这种平等和一致性并不仅限于 Java [开发工具](http://baike.so.com/doc/6917134-7139022.html)。尽管 Eclipse 是使用Java语言开发的，但它的用途并不限于 Java 语言;例如，支持诸如C/C++、[COBOL](http://baike.so.com/doc/5446502-5684869.html" \t "_blank)、PHP、[Android](http://baike.so.com/doc/7090794-7313709.html" \t "_blank)等[编程语言](http://baike.so.com/doc/538266-569890.html" \t "_blank)的插件已经可用，或预计将会推出Eclipse 框架还可用来作为与软件开发无关的其他应用程序类型的基础，比如[内容管理系统](http://baike.so.com/doc/4020997-4218290.html" \t "_blank)。

基于 Eclipse 的应用程序的一个突出例子是 IBM Rational Software Architect，它构成了 IBM Java 开发工具系列的基础。

**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**](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**

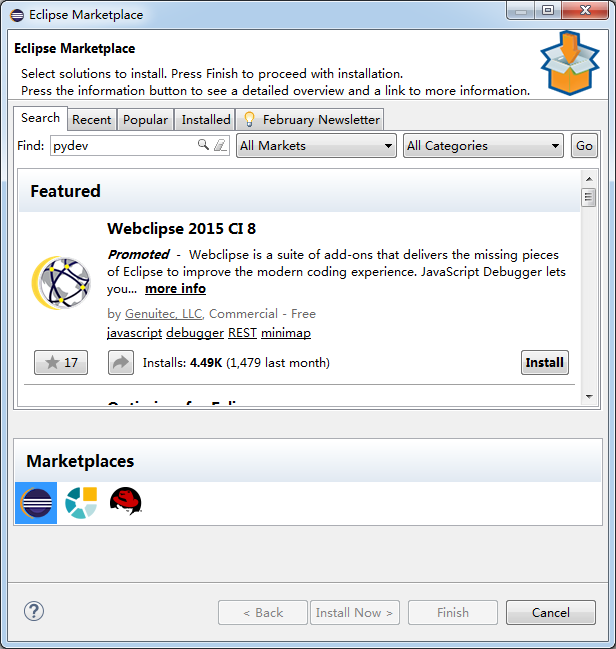
使用Eclipse IDE作为Python开发环境，需要：

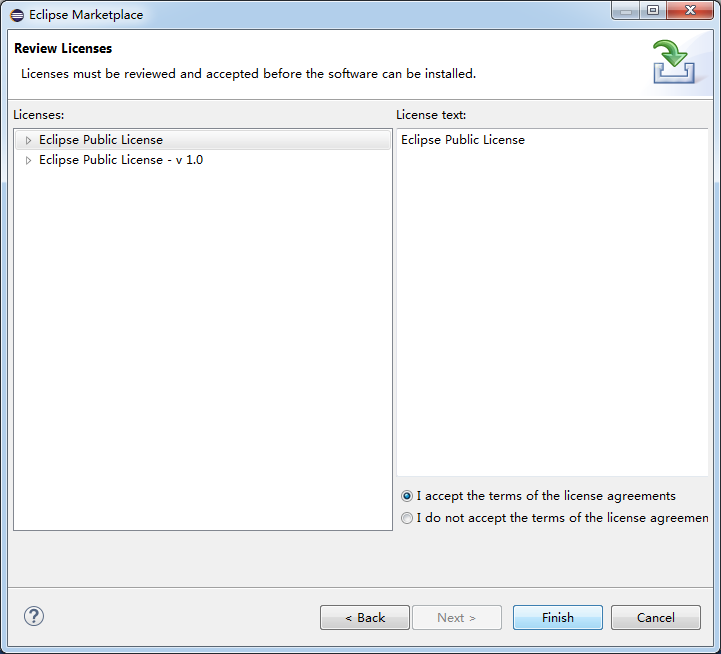
1） 安装PyDev插件；

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

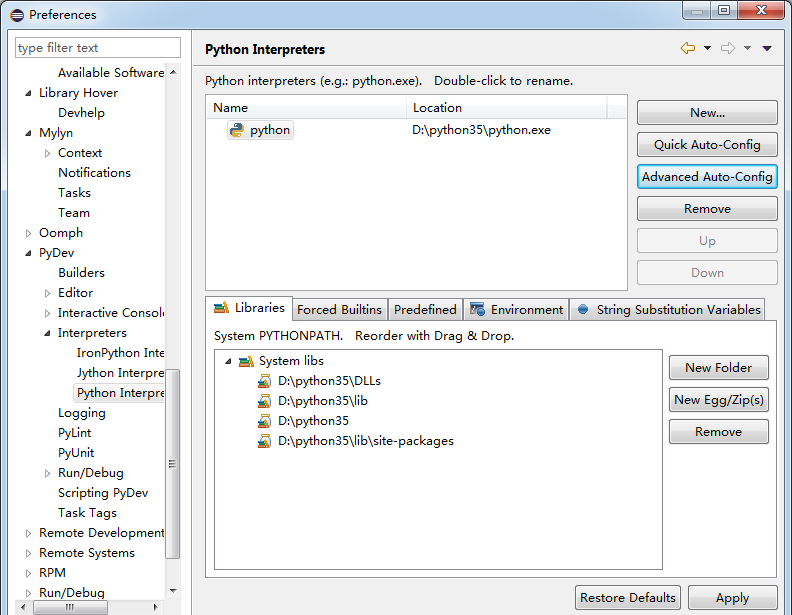
离线安装.



**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。

**3.3 联机指南**

**Help** -> **Help Contents**



**四 配置 PyDev提高效率**

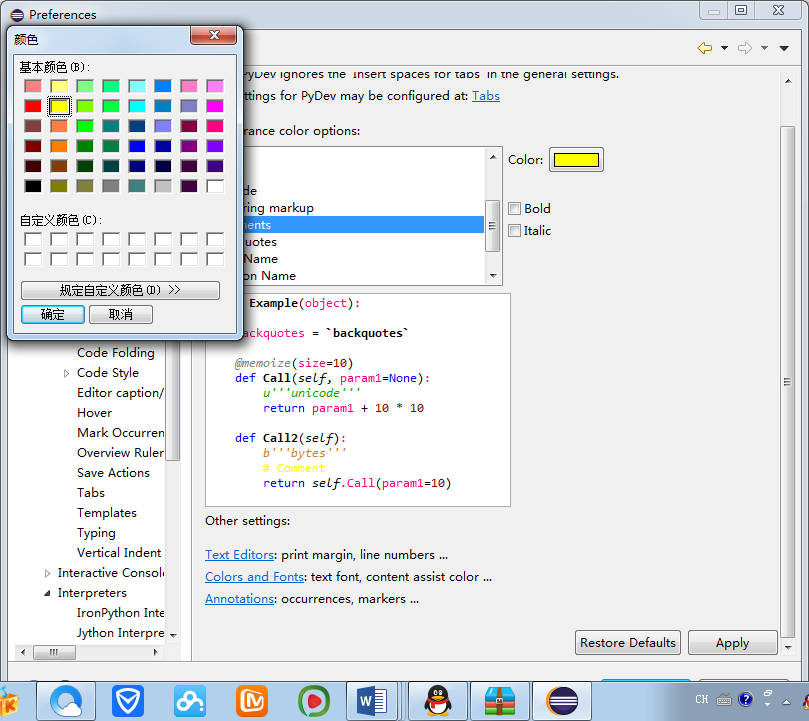
**4.1 显示源码行号**

右键源码的左边缘，选中“Show Line Number”

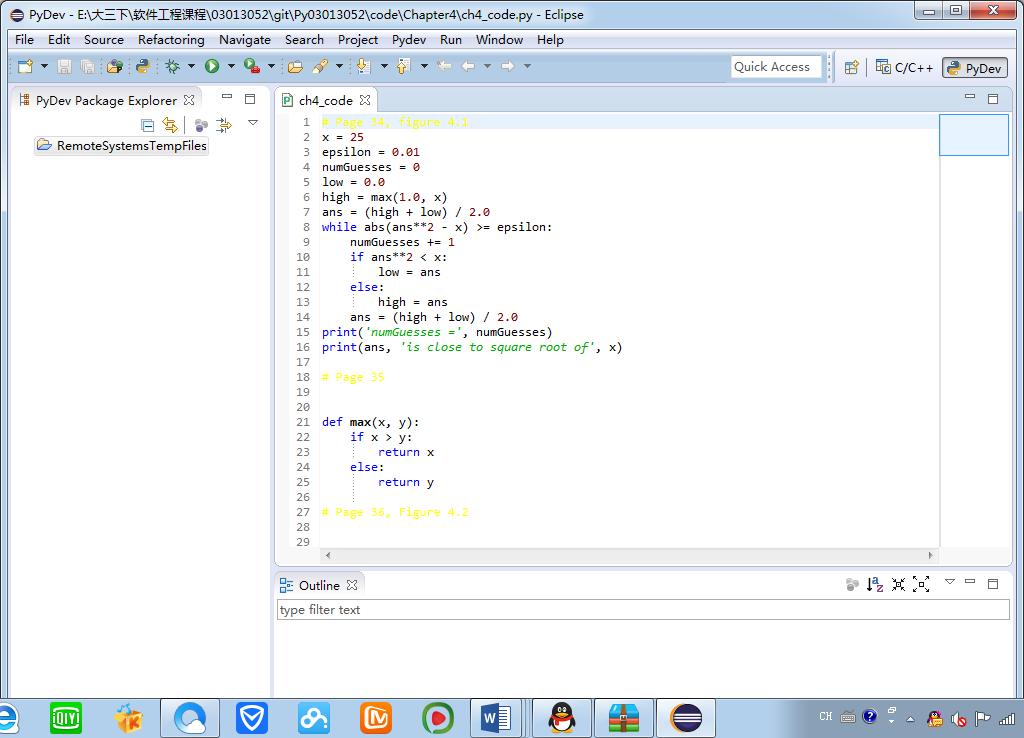


**4.2 修改注释颜色提高可读行**

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



颜色修改后，显示如下图。

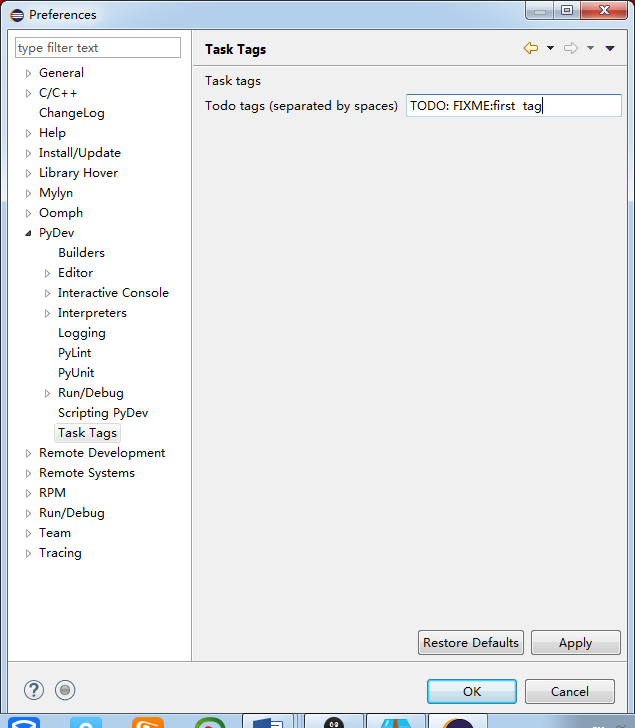


**4.3 配置任务标签**

在程序开发过程中可以在代码中标识，当前任务状态，计划开发工作。在代码中标识任务，可以使用任务标签，然后，让开发环境识别到工作空间的任务列表中。

1. **配置任务标签：**

通过PyDev->Task Tags中配置任务标签:



可以根据不同的任务类型，自己增加自定义标签

**2）添加任务标签注释**

在程序源码中加入使用任务标签的注释：



**3）识别任务**

保存新修改、运行程序一次或者选择 Project → Clean

就可将使用任务标签注释加入任务列表。如果当前任务窗口可视，刚加的任务，就会立刻显示在任务窗口中。



如任务窗口没有打开，可以Window->Show View->Tasks开启任务窗口。

**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。

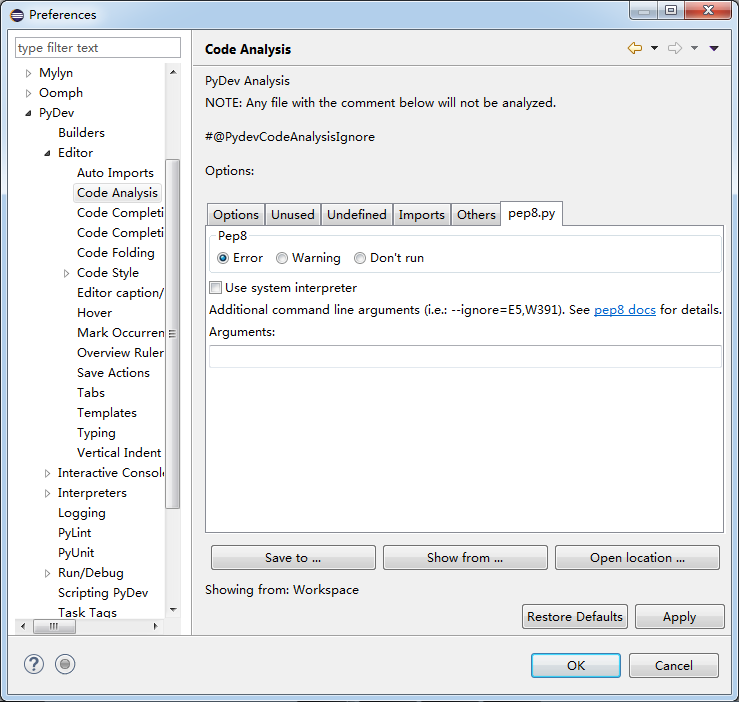
程序开发过程中，要有规范意识，但不可能有很高的规范性，过分注意规范会影响开发进程。这时如果一直开启代码规范检查，经常提示不规范，会对开发形成负面影响，所以，默认关闭是合适的。在程序开发一个阶段结果出来时，进行规范性检查更好。

**5.1 PEP8检查和修改**

**5.1.1 启动 pep8 检查**

Window > Preferences PyDev > Editor > Code Analysis > pep8.py

选择Errors/Warnings其中之一..



右键Python工程，选择 PyDev，点 "code analysis"，即可对工程中所有Python源码进行PEP8检查： .



**5.1.2 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?

|  |
| --- |
|  |

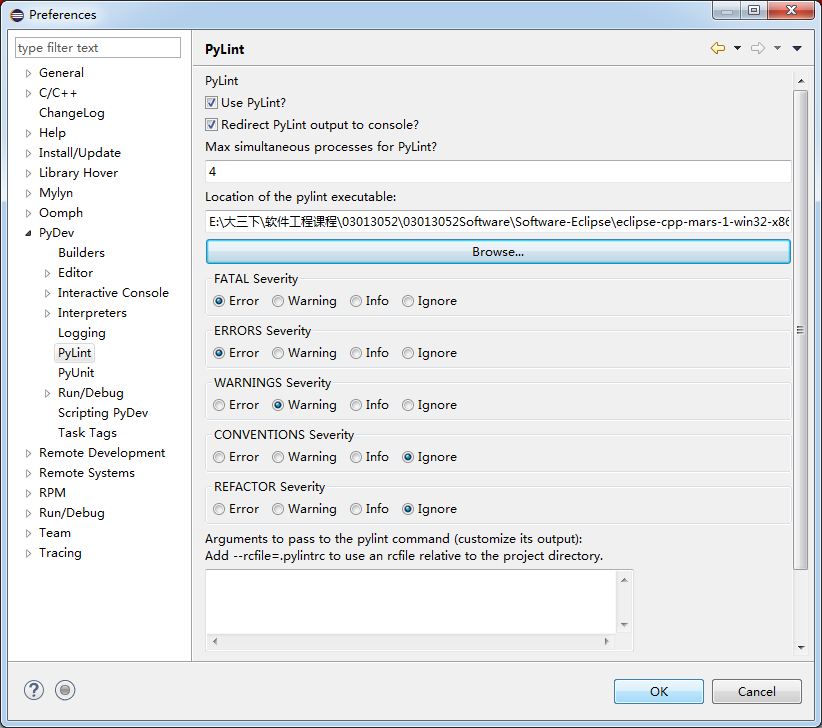
在Python源码窗口，按 CTRL-SHIFT-F 就可以自动修改代码

**5.2 使用Pylint**

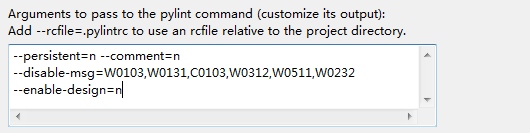
PyDev 默认不开启Pylint。通过

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

找到安装好的lint.py的地址,例如"C:\Python34\Lib\site-packages\pylint\lint.py"



配置参数，限制Pylint的输出



选中Project->Build Automatically,这样程序修改，保存时pylint就会自动检查项目中的代码,也可用Ctrl+B手动build触发pylint。



**小结**

Python是一种简单方便且应用广泛的语言，但是作为一名初学者，我对这门语言的掌握实在是很少，希望通过软件工程这门课程我能在这一学期中多多了解它，从而也能让自己掌握更多关于软件学习的知识。

**参考文献**

1．Brainwy Software Ltd. PyDev Manual. http://www.pydev.org/manual\_101\_root.html

2．郑伟芳. PyDev for Eclipse简介. <http://www.ibm.com/developerworks/cn/opensource/os-cn-ecl-pydev/> 2008.11

3．张颖. Python 代码调试技巧. <http://www.ibm.com/developerworks/cn/linux/l-cn-pythondebugger/> 2012.05