|  |
| --- |
|  |
| Python开发环境建立 |
| 指导手册 |
|  |
| **03013302 李婷** |
| **2016/3/21** |

[**一、** **Python基本开发环境建立** 2](#_Toc446401534)

[**1.1 Python简介** 2](#_Toc446401535)

[**1.2 开发环境建立** 2](#_Toc446401536)

[**二、** **Python扩展包安装** 3](#_Toc446401537)

[**2.1 编码规范** 3](#_Toc446401538)

[**2.2 科学计算软件包** 3](#_Toc446401539)

[**2.2.1 SciPy** 3](#_Toc446401540)

[**2.2.2 交互计算 Jupiter** 5](#_Toc446401541)

[**2.2.3 IF97 for Python** 6](#_Toc446401542)

[**三、** **基于Eclipse的Python开发环境** 6](#_Toc446401543)

[**3.1安装eclipse IDE** 6](#_Toc446401544)

[**3.1.1安装Java** 6](#_Toc446401545)

[**3.1.2 Eclipse IDE** 7](#_Toc446401546)

[**3.2 安装Python开发插件PyDev** 8](#_Toc446401547)

[**3.2.1 安装PyDev插件** 8](#_Toc446401548)

[**3.2.2 配置 Python解释器** 9](#_Toc446401549)

[**四、** **配置 PyDev提高效率** 10](#_Toc446401550)

[**4.1 显示源码行号** 10](#_Toc446401551)

[**4.2 修改注释颜色提高可读行** 10](#_Toc446401552)

[**4.3 配置任务标签** 10](#_Toc446401553)

[**4.3.1配置任务标签：** 11](#_Toc446401554)

[**4.3.2添加任务标签注释** 11](#_Toc446401555)

[**4.3.3识别任务** 11](#_Toc446401556)

[**五、** **基于PyDev的语言规范静态检查** 12](#_Toc446401557)

[**5.1 PEP8检查和修改** 12](#_Toc446401558)

[**5.2 使用Pylint** 12](#_Toc446401559)

1. **Python基本开发环境建立**

**1.1 Python简介**

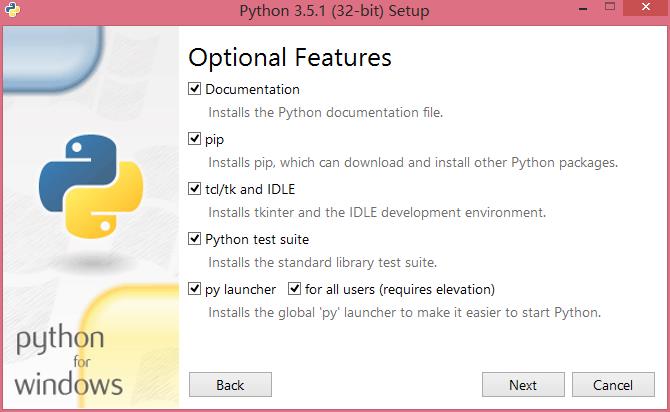
开发环境Python, 是一种面向对象、解释型计算机程序设计语言，由Guido van Rossum于1989年发明，第一个公开发行版发行于1991年。Python是纯粹的自由软件， 源代码和解释器CPython遵循 GPL(GNU General Public License)协议。

Python语言的优点有：（1）功能强大，使用方便。具有资源丰富、涉及范围广泛的第三方库/模块 （2）可移植性 （3）python语言核心的库是开放的 （4）可嵌入性

**1.2 开发环境建立**

官网：<http://www.python.org>

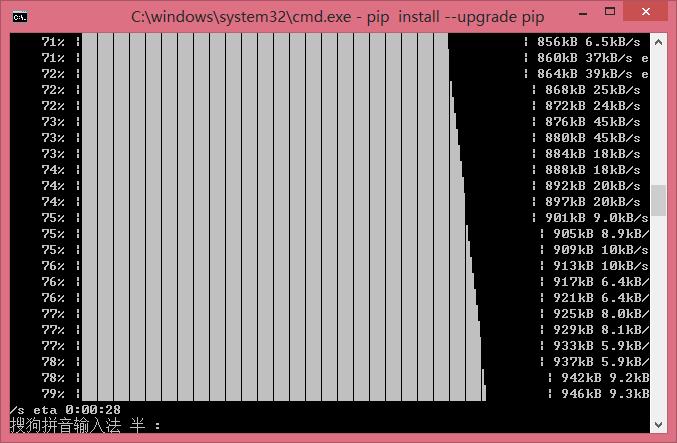
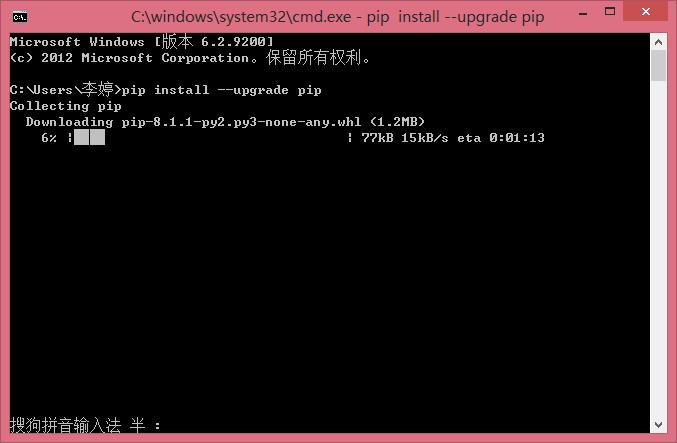
从python官网地址选择python3.5.1版本下载，我的电脑是64位Windows操作系统，安装时注意加环境变量path。我选择在C盘新建文件夹python35来下载安装软件。

安装的32位版本时候的截图，后来又重新删除安装了和操作系统对应的64位版本。

然后，命令行下更新pip到新版本：

>pip install --upgrade pip



如果在更新过程中出现失败，需要卸载Python，删除一切相关文件，重新安装之后在命令行下输入：

>python –m pip install --upgrade pip

1. **Python扩展包安装**

**2.1 编码规范**

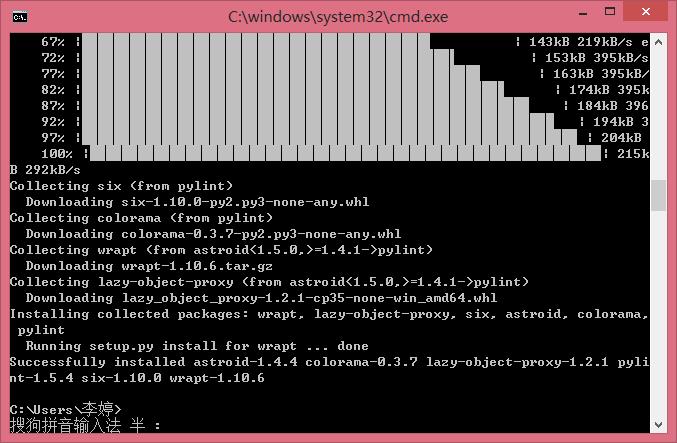
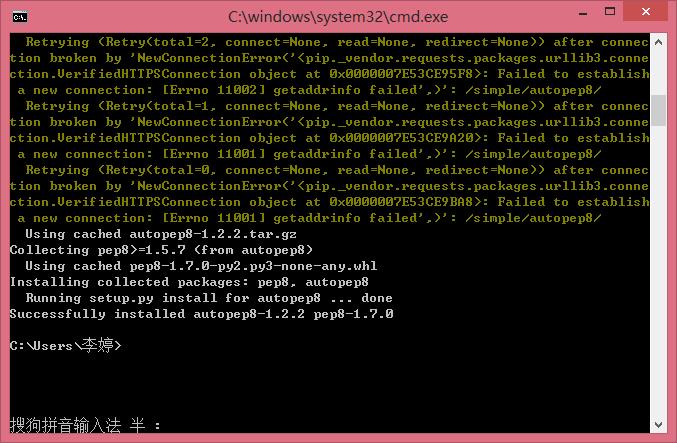
Autopep8是一个将Python代码自动排版为PEP8风格的小工具。它使用pep8工具来决定代码中的哪部分需要被排版。Autopep8可以修复大部分pep8工具中报告的排版问题。

Pylint 是一个 Python 代码分析工具，它分析 Python 代码中的错误，查找不符合代码风格标准和有潜在问题的代码。除此之外，Pylint还可以提供其他的功能，如：检查一行代码的长度，变量名是否符合命名标准，一个声明过的接口是否被真正实现等等。Pylint 的高可配置性，高可定制性是最突出的优点，而且可通过写小插件来添加功能。改进代码之后再运行，会同时显示出两次的运行结果，从而让操作人员对代码质量有直观的对比。

在线安装即在命令行下输入：

>pip install autopep8

>pip install pylint



**2.2 科学计算软件包**

**2.2.1 SciPy**

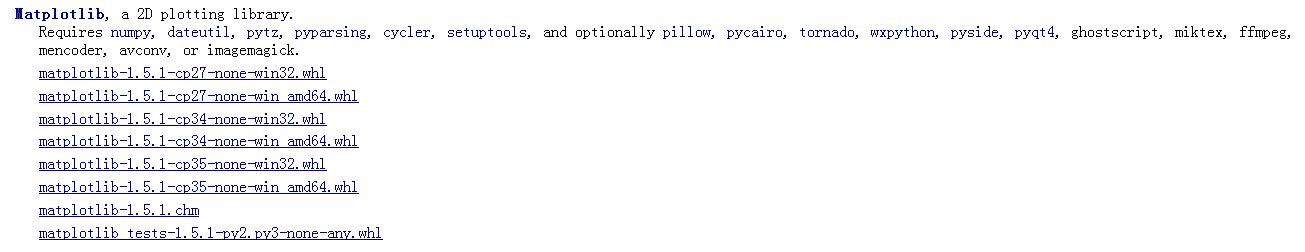
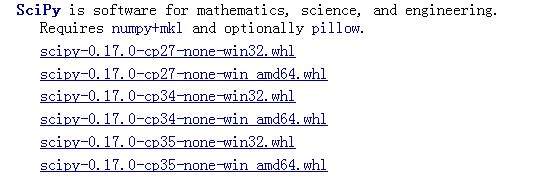
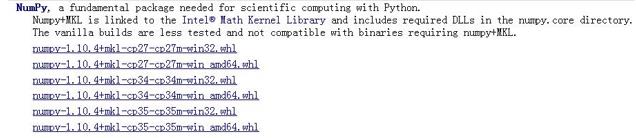
SciPy是一款方便、易于使用、专为科学和工程设计的Python工具包.它包括统计,优化,整合,线性代数模块,傅里叶变换,信号和图像处理,常微分方程求解器等等。

在加州大学欧文分校提供的Windows32/64位Python扩展包编译版，

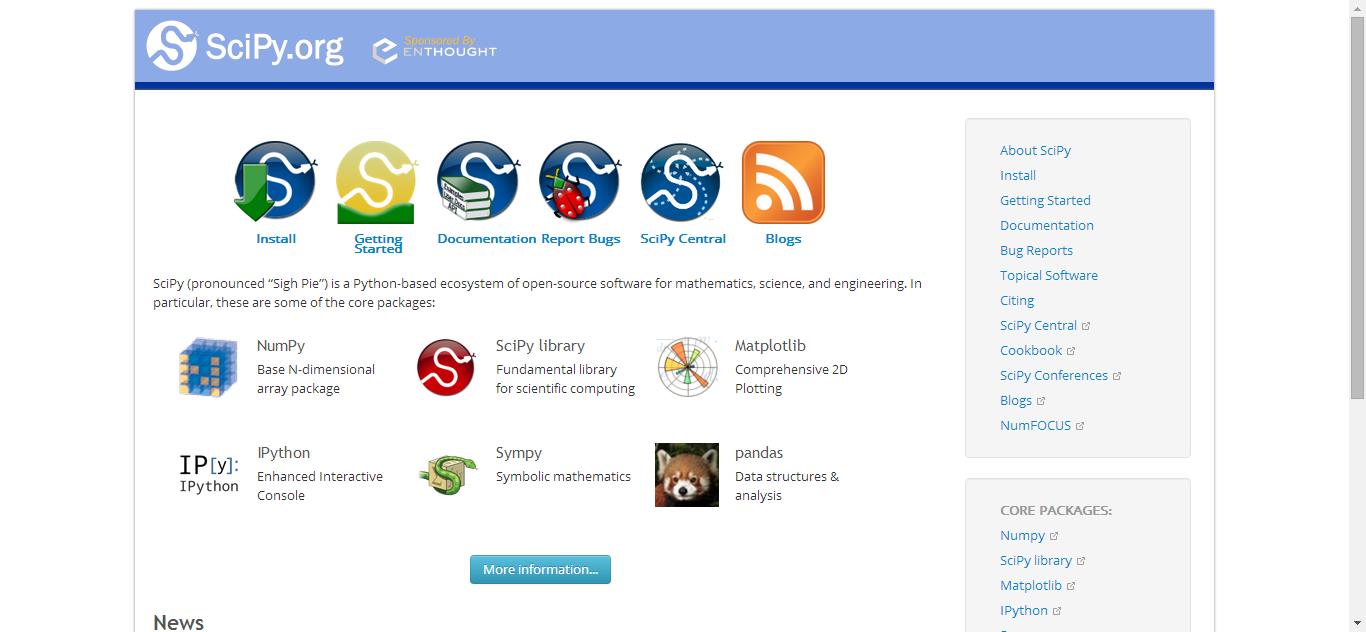
http://www.lfd.uci.edu/~gohlke/pythonlibs/

可以方便查到到科学计算软件,然后，用pip逐个本地安装。

>pip install \*.whl



也可以从Scipy官网：<http://www.scipy.org/>直接安装下载。



>pip install numpy

>pip install scipy

>pip install matplotlib

**2.2.2 交互计算 Jupiter**

1. 安装Jupiter ：

>pip install jupyter 在线安装

1. 安装Python语言内核

>pip install ipython ，支持Python语言

1. 安装依赖包

>pip install pyreadline

>pip install sympy

以上三项内容安装方法基本一样，不再重复说明。

1. 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

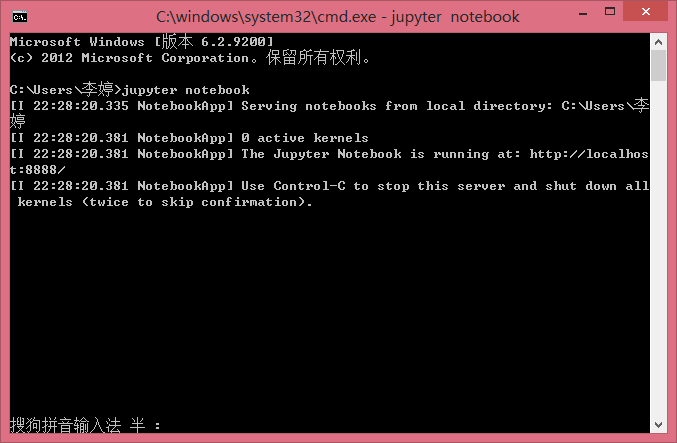
install\_mathjax()

因为新版本jupyter已经默认安装了mathjax，现在不需要以上安装过程了。

1. 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

1. **基于Eclipse的Python开发环境**

选用理由：Eclipse是由IBM提供的一个开源的Java程序开发软件。它可以为快速开发Java程序提供集成的开发环境（IDE）。编辑、构造、调试和在线帮助被集成在一个用户图形界面。有效的使用可以极大的提高编程速度。

**3.1安装eclipse IDE**

**3.1.1安装Java**

Eclipse IDE是使用Java开发的,首先需要从Java的官方网址下载，地址如下：

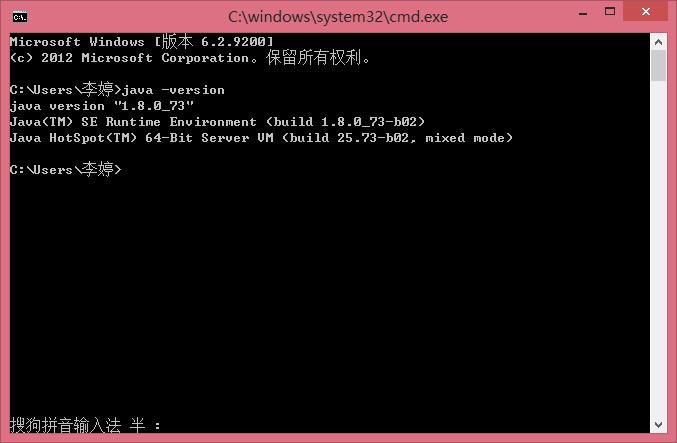
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>



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**。**

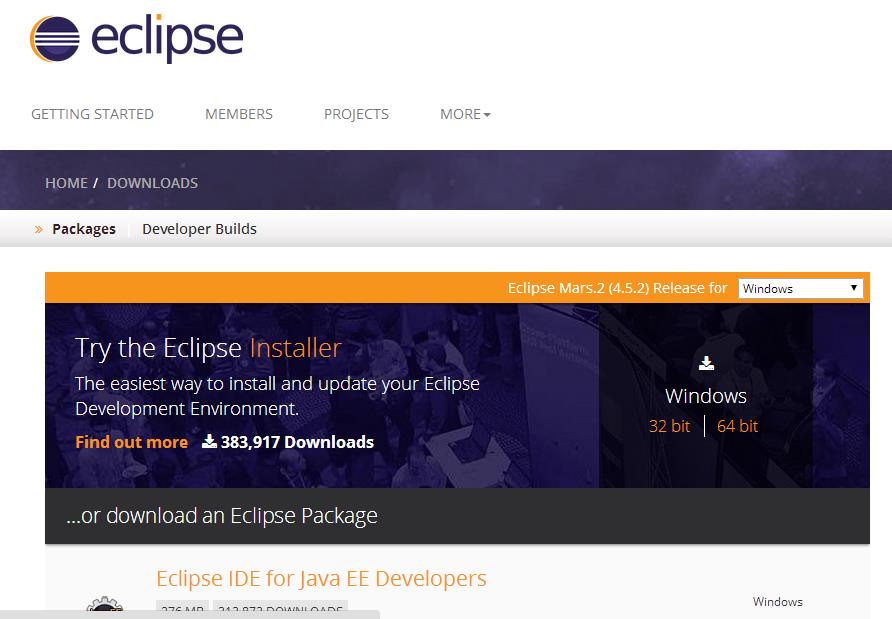


**3.1.2 Eclipse IDE**

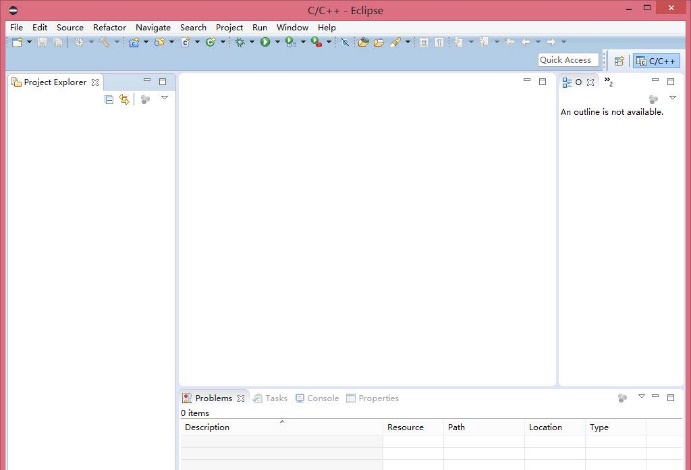
　　Eclipse IDE是插件型开发环境，有很多版本可以下载。本课程主要使用Python语言，会涉及C/C++开发，所以，建议下载Eclipse CDT(Eclipse IDE　for　C/C++)版.

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。安装成功之后固定到任务栏，之后打开使用会很方便。打开界面如下：

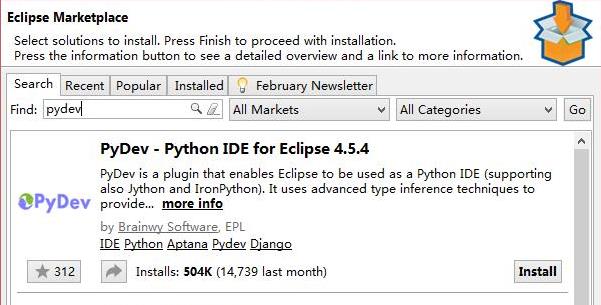


**3.2 安装Python开发插件PyDev**

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**

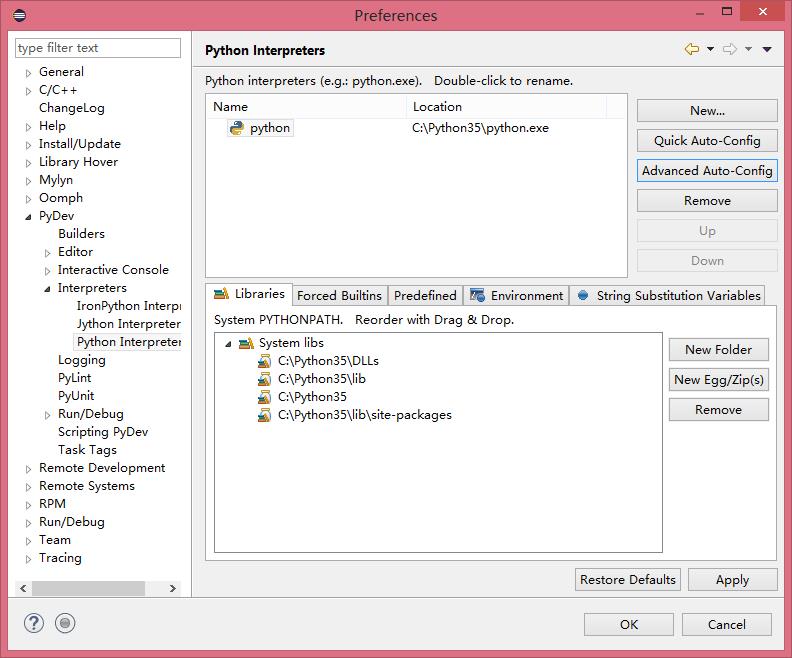
通过Help->Eclipse Marketplaces进入市场，输入Pydev，找到Pydev安装/更新项目，在线安装即可：



在线安装过程简单，但是，安装速度受网络环境影响。

**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。

1. **配置 PyDev提高效率**

**4.1 显示源码行号**

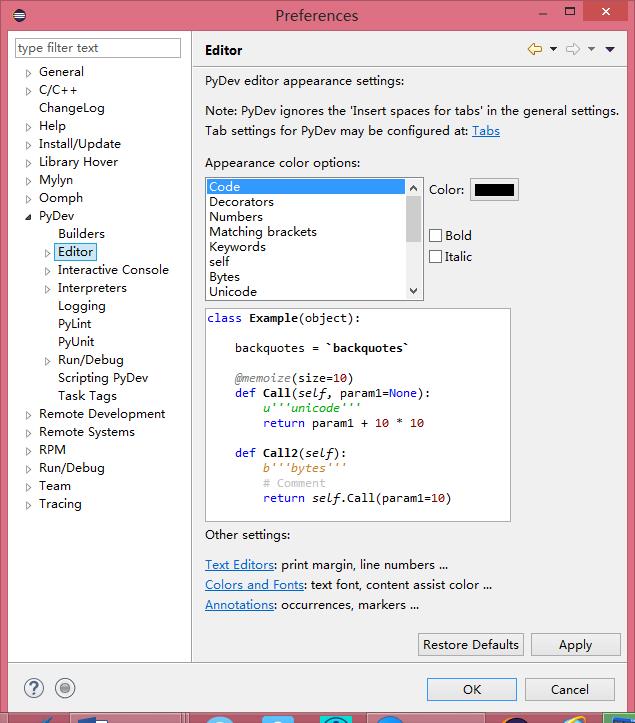
右键源码的左边缘，选中“Show Line Number



因为我是使用浏览器截图，截图时该选项框自动消失，所以用的是课件配图。

**4.2 修改注释颜色提高可读行**

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

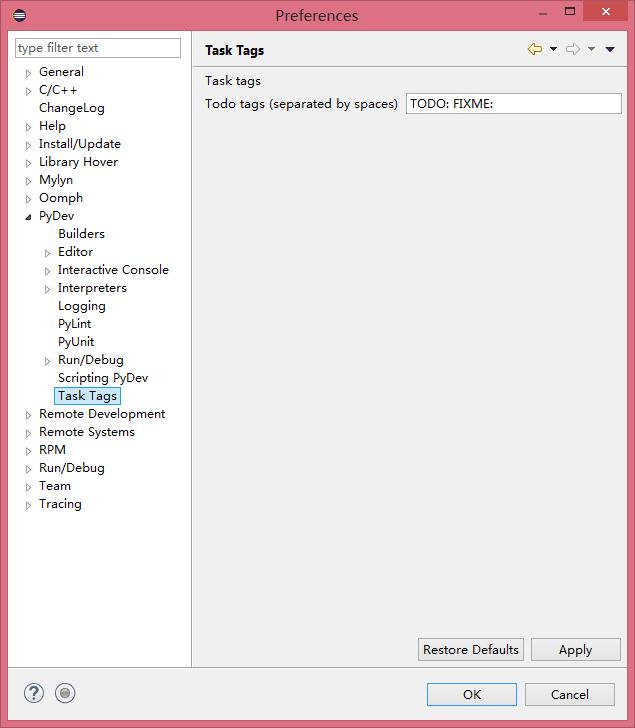


**4.3 配置任务标签**

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。

**4.3.1配置任务标签：**

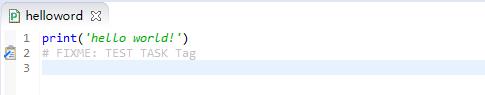
通过PyDev->Task Tags中配置任务标签:



可以根据不同的任务类型，自己增加自定义标签

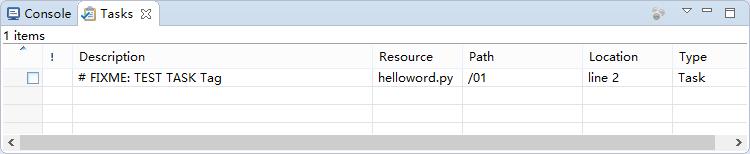
**4.3.2添加任务标签注释**

在程序源码中加入使用任务标签的注释：



**4.3.3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。

1. **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。

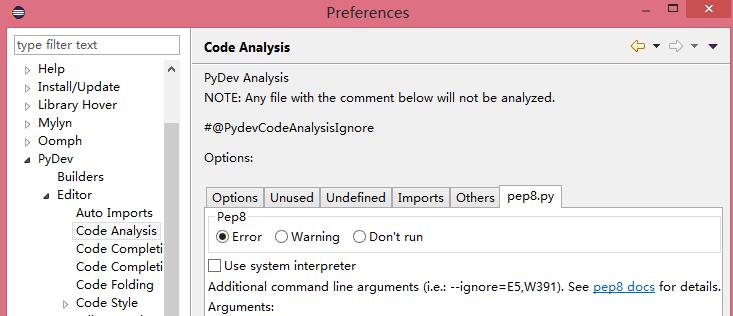
**5.1 PEP8检查和修改**

1）启动 pep8 检查：

Window > Preferences

PyDev > Editor > Code Analysis > pep8.py

选择Errors/Warnings其中之一。



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。

1）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

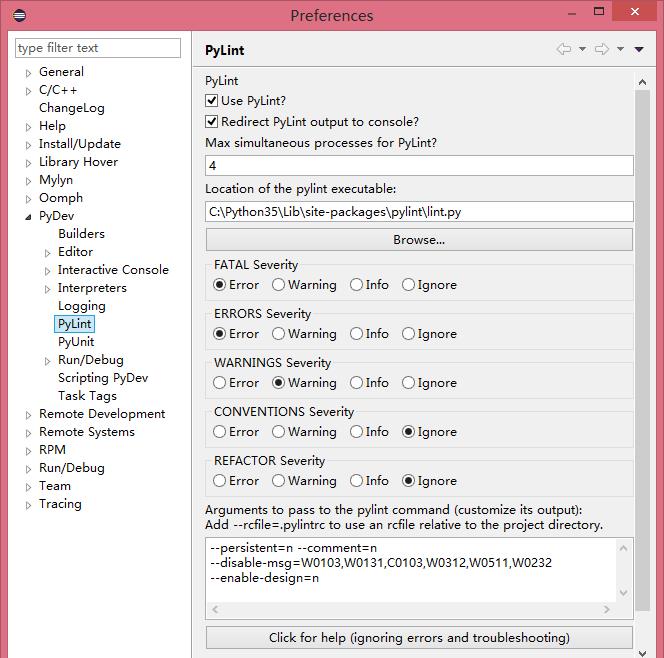
**5.2 使用Pylint**

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：

C:\Python35\Lib\site-packages\pylint\lint.py"



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。