**环境建立指导手册**

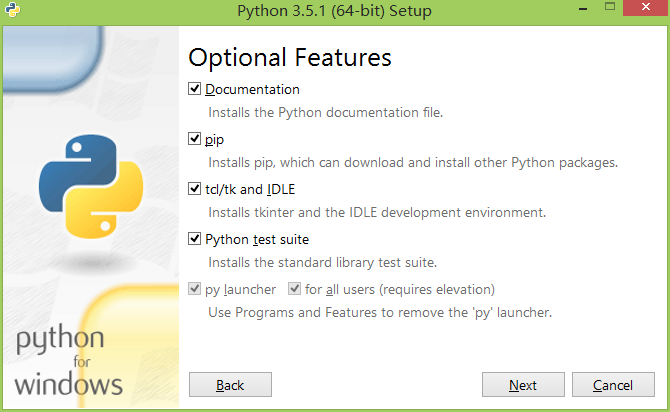
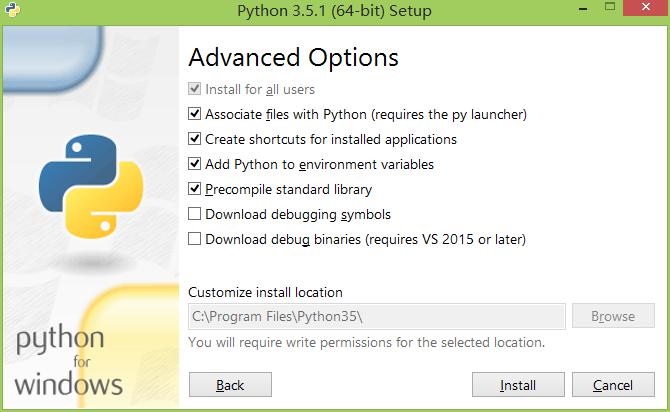
Python解释器、标准库及扩展包构成Python的基本开发环境，满足学习和软件开发的要求。Python解释器及其标准库可从Python官网下载；其他扩展包可用pip命令等方法安装。

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。

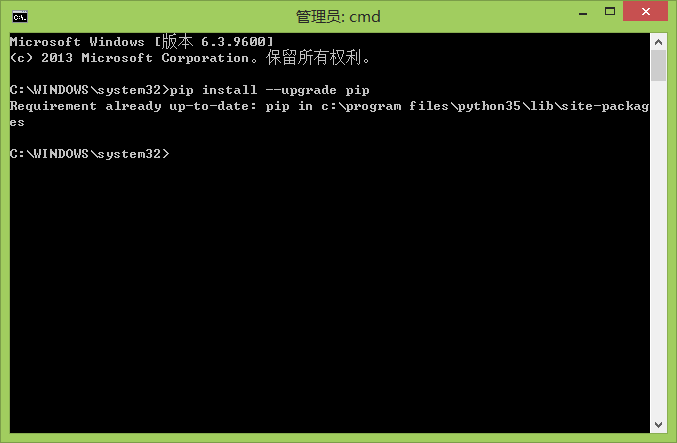
1. **Python基本开发环境建立**

**1.1** **Python基本开发环境建立**

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。Windows系统安装python时,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（由于安装在C:\python35时无法安装tcl/tk and IDLE，故将安装目录定为C:\Program Files\python35）

然后，命令行下更新pip到新版本：>pip install --upgrade pip



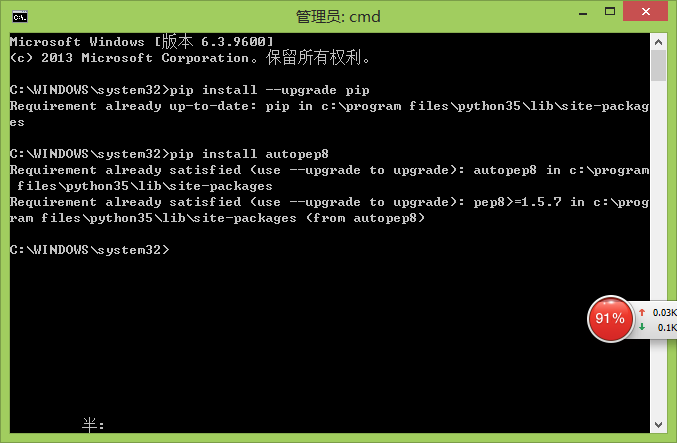
1. **安装扩展包**

**2.1 安装格式规范扩展包**

**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。Autopep8可以修复大部分pep8工具中报告的排版问题。

安装命令：pip install autopep8



**2.1.2 pylint**

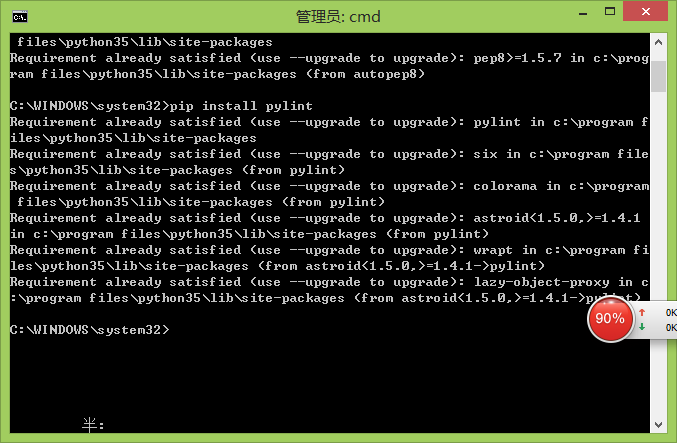
Pylint 是一个 Python 工具，除了平常代码分析工具的作用之外，它提供了更多的功能：如检查一行代码的长度，变量名是否符合命名标准，一个声明过的接口是否被真正实现等等。

Pylint 的一个很大的好处是它的高可配置性，高可定制性，并且可以很容易写小插件来添加功能。

如果运行两次 Pylint，它会同时显示出当前和上次的运行结果，从而可以看出代码质量是否得到了改进。

目前在 eclipse 的 pydev 插件中也集成了 Pylint。

安装命令：pip install pylint



**2.2 科学计算软件包**

**2.2.1 SciPy**

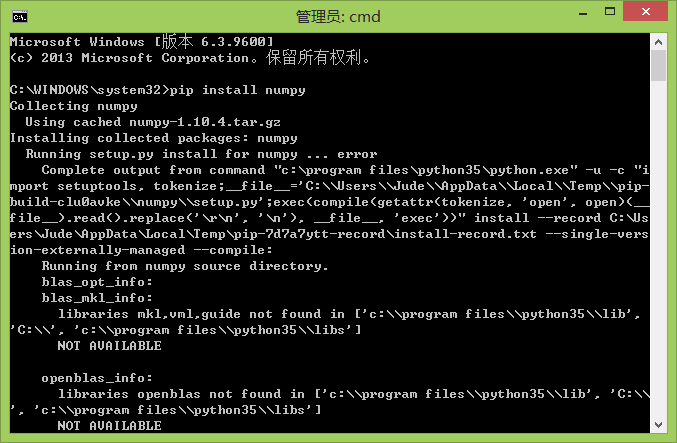
SciPy是一款方便、易于使用、专为科学和工程设计的Python工具包.它包括统计,优化,整合,线性代数模块,傅里叶变换,信号和图像处理,常微分方程求解器等等。

安装Scipy中的3个扩展包：

pip install numpy

pip install scipy

pip install matplotlib

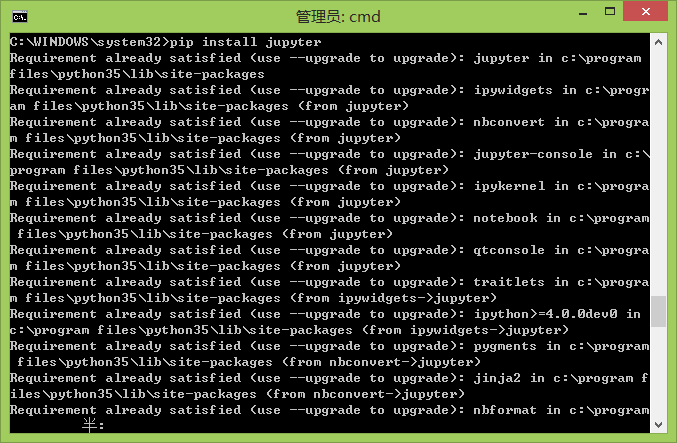


由于其余两扩展包的安装方式相同，故不再截图演示。

**2.2.2 交互计算 Jupiter**

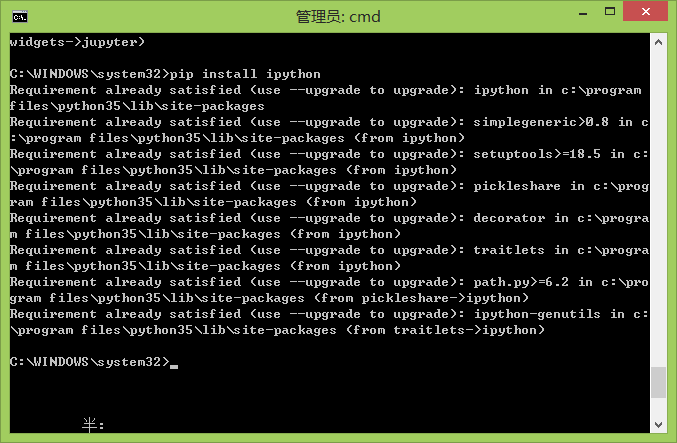
1） 安装Jupiter ：

>pip install jupyter



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

由于其余两扩展包的安装方式相同，故不再截图演示。

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

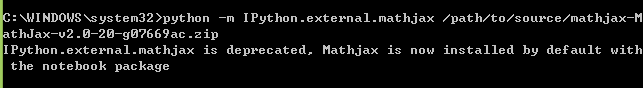
install\_mathjax()

由于用此方法进行在线安装失败，故在百度中找到此网站

<http://www.throwexcept.com/article/1418122387791.html>

用离线安装法进行安装，在cmd中输入命令：

python -m IPython.external.mathjax /path/to/source/mathjax-MathJax-v2.0-20-g07669ac.zip



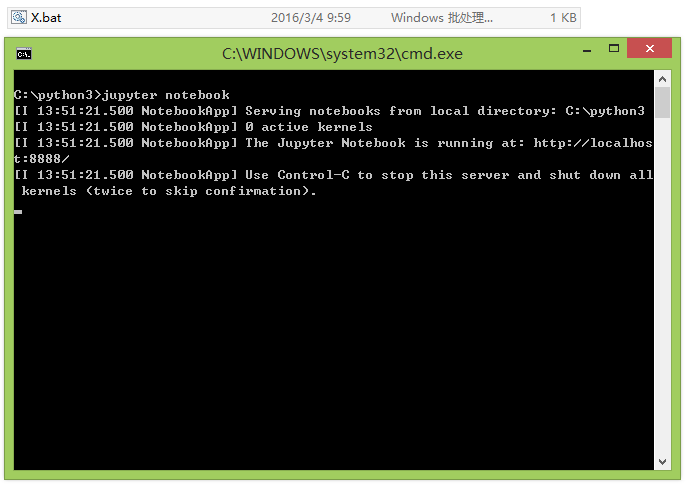
由于为安装后进行补截图，故仅有安装成功后的记录。

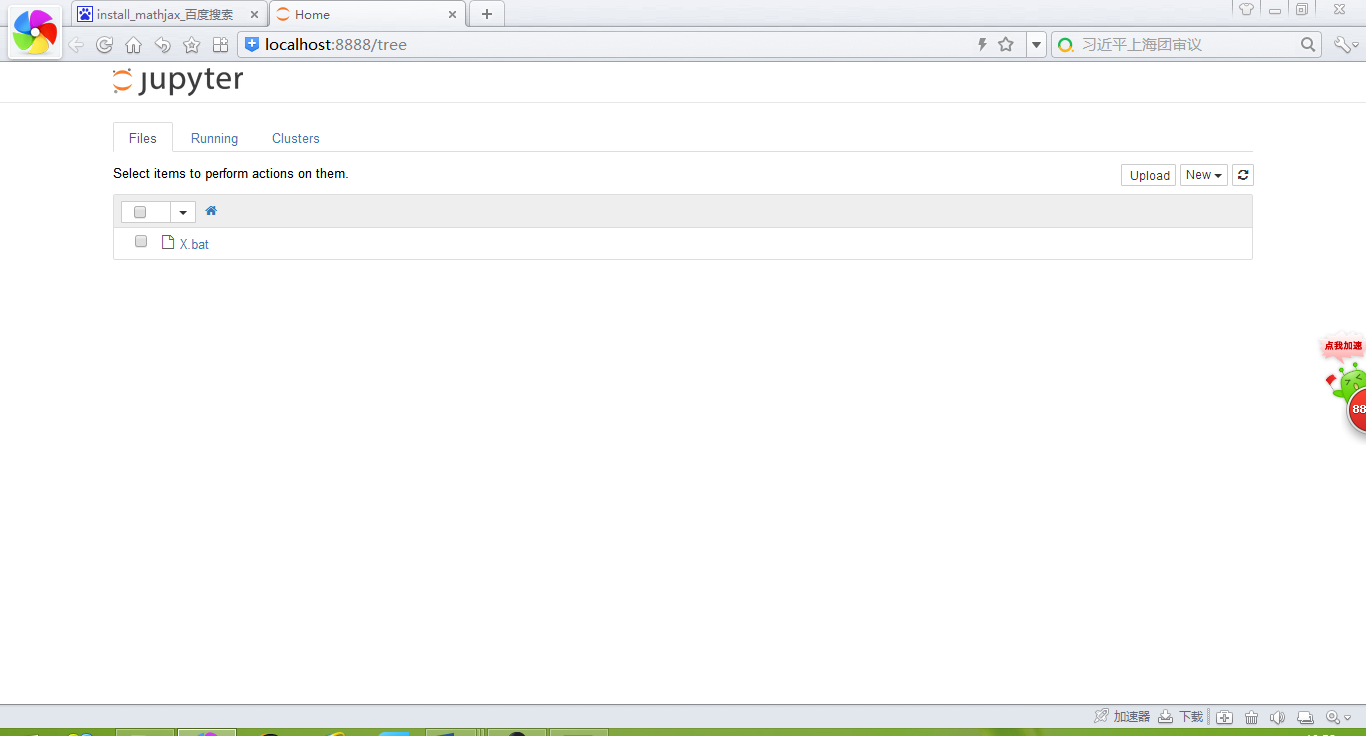
5） 运行notebook:

在iPython notebook文件所在目录下，打开命令行窗口：

>jupyter notebook

也可通过创建.Bat文件进行快捷呼出。





**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

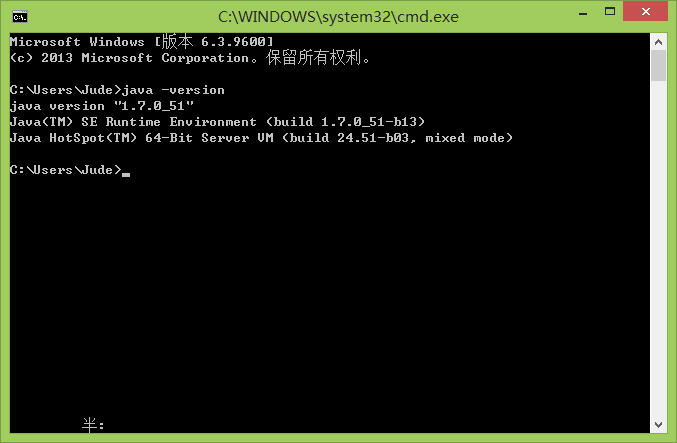
1. **基于Eclipse的Python开发环境**

Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台，专注于为高度集成的工具开发提供一个全功能的、具有商业品质的工业平台。Eclipse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。Eclipse 附带了一个标准的插件集，包括 Java 开发工具（Java Development Tools，JDT）。Eclipse 开源，而且一般Java开发更倾向于Eclipse，使用安装免费，操作简单，且包含很多插件，使用方便，但程序启动较慢。

**3.1 安装Eclipse IDE**

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。



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32/64位，下载相应的版本，然后将下载的Eclipse CDT解压到指定目录下，并将eclipse.exe固定到任务栏。

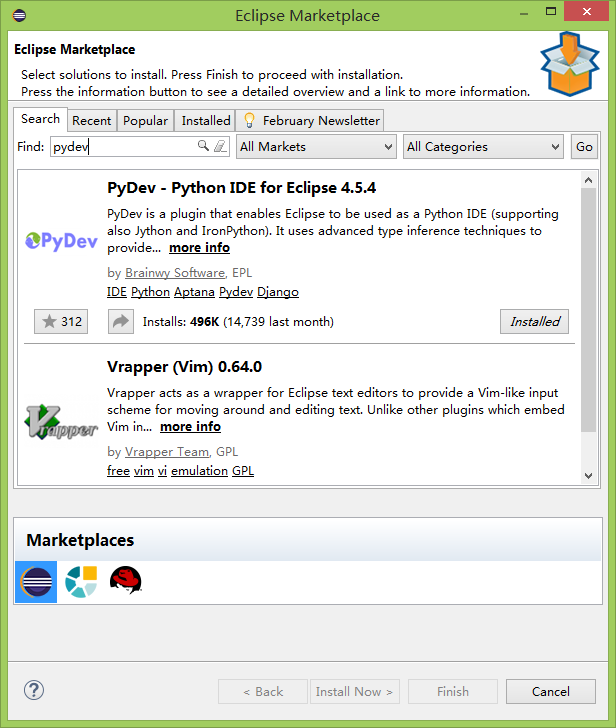
注：eclipse为解压即用文件，不需要进行安装。



**3.2 安装Python开发插件PyDev**

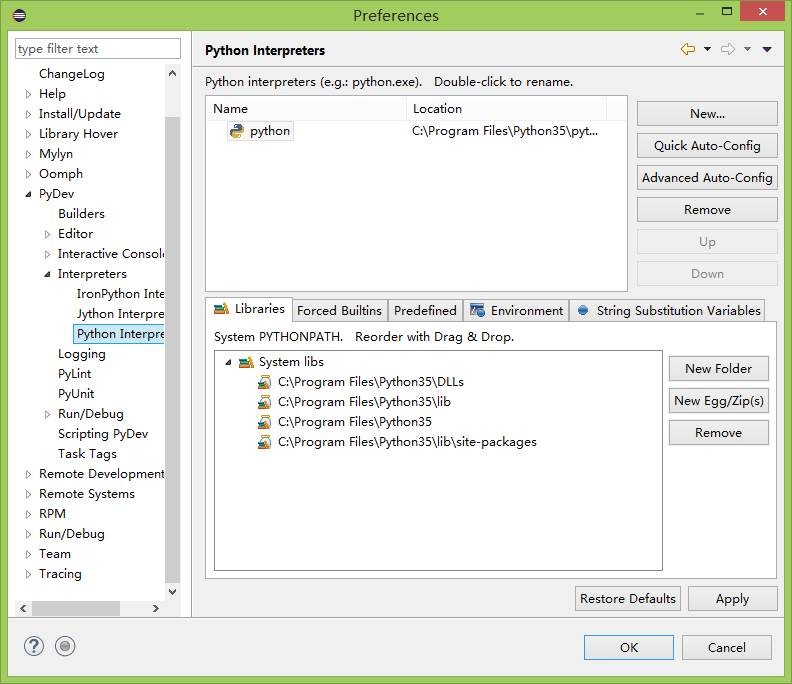
使用Eclipse IDE作为Python开发环境，需要：1） 安装PyDev插件；2）配置解释器为Python3.5；

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：



安装好后，重新启动。通过：

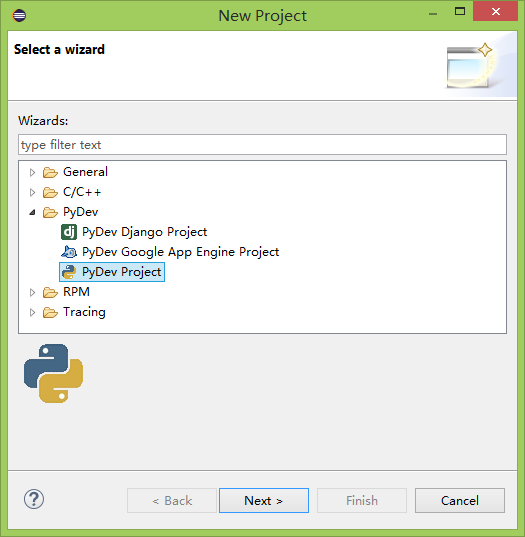
Windows->Preference->Pydev->Interperters->Python Interperter点其中的：Advanced Auto-config配置开发使用的 Python解释器版本:



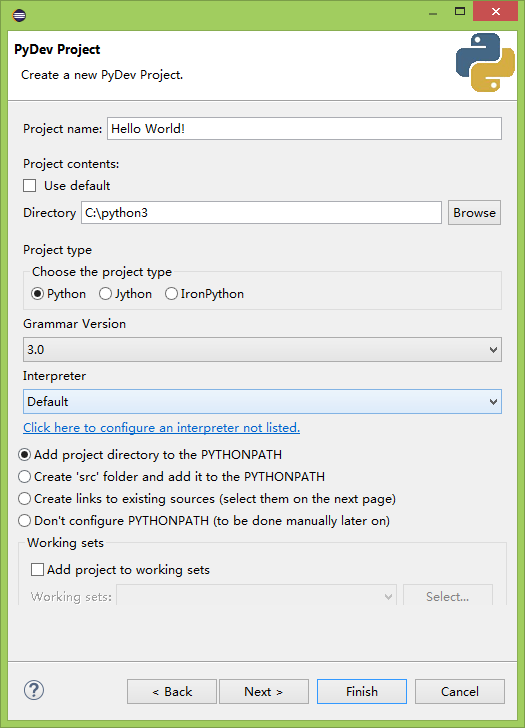
配置好后切换到Python场景，就可以进行Python开发了。

**3.3 创建python工程的方法**

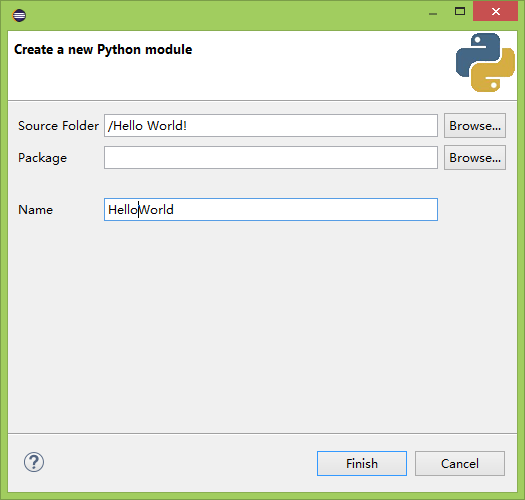
通过File->New->Project，在New Project中选择Pydev->Pydev Project，建立新的工程



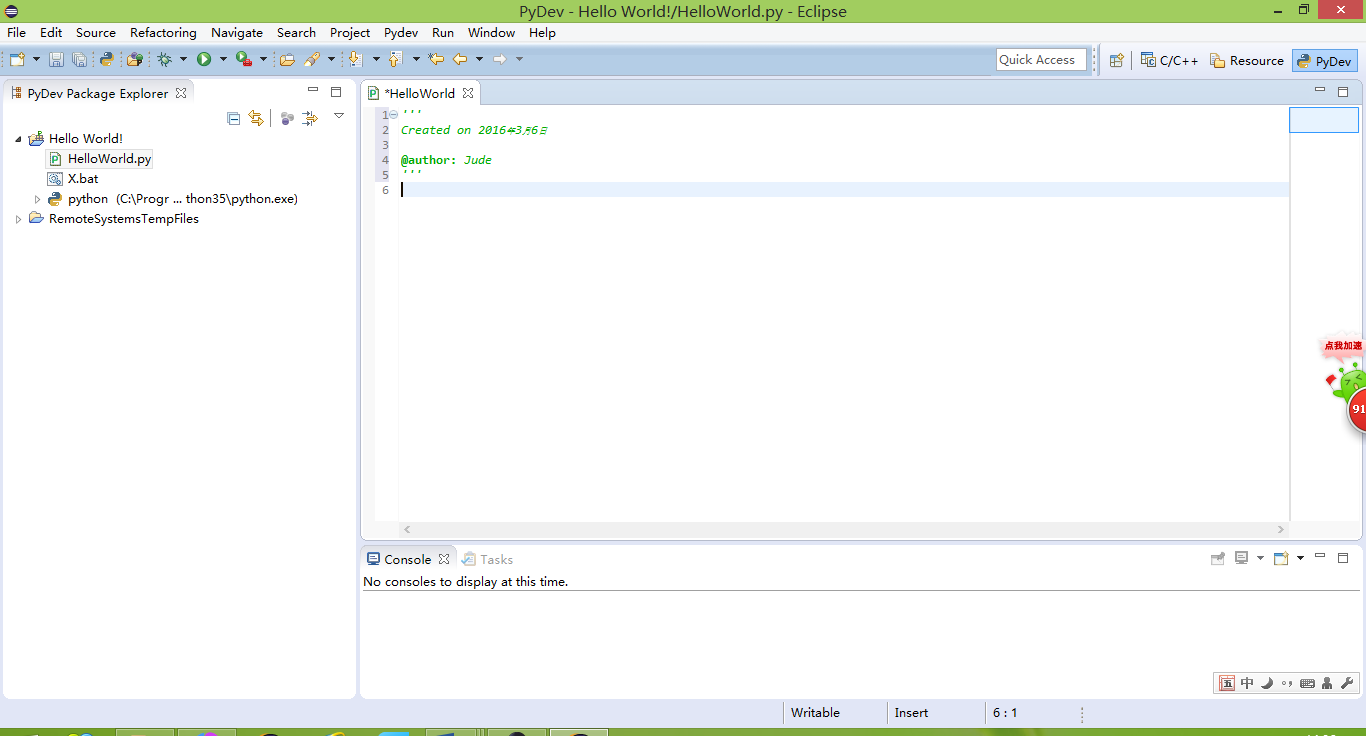
选择语法版本为3.0，取消使用默认保存位置，改为如图所示位置。



在左侧workplace中选择新建的工程，File->New->Pydev Module以建立可编写窗口。（注：文件名Name中不能有空格）



将已有的模版内容删除即可做为空工程使用。

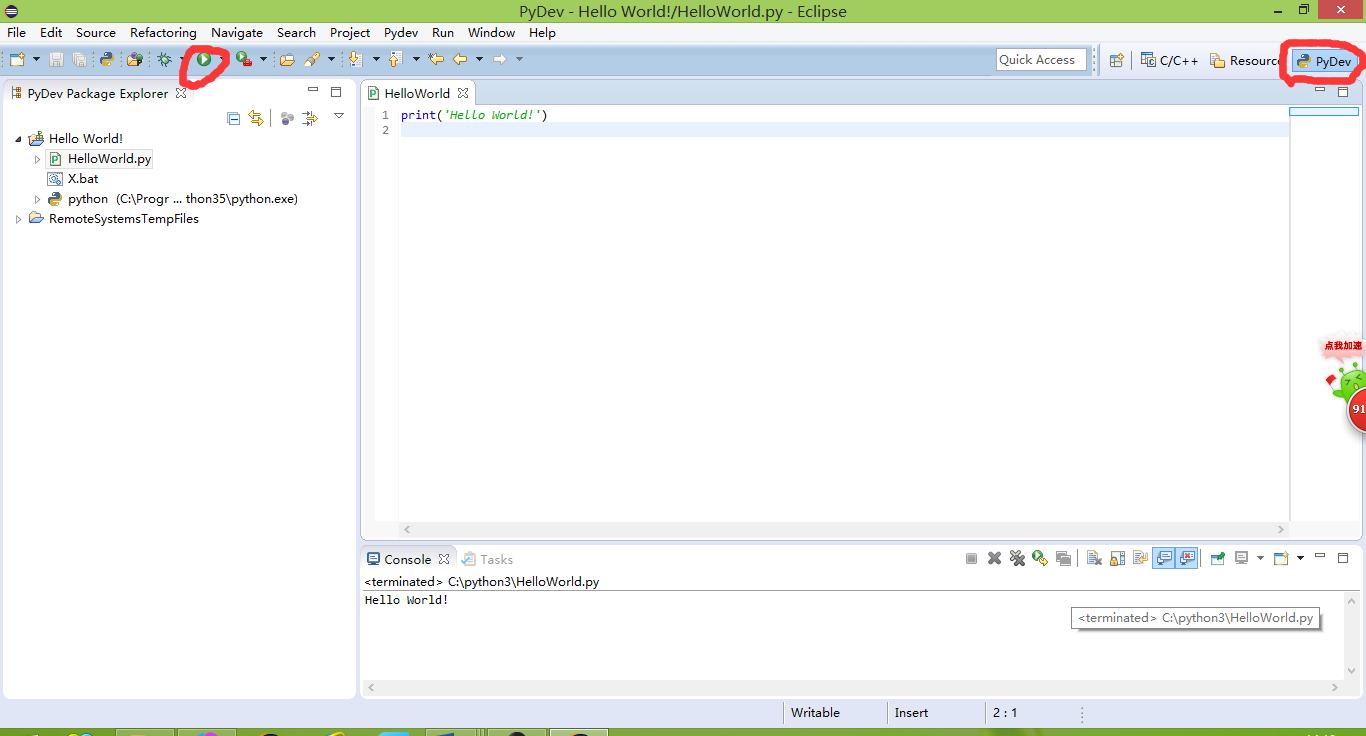


编写最简单的工程示例Hello World!作为示范

代码内容：

print(*'Hello World!'*)

于右上角选择PyDev，并点击左侧的运行按钮运行代码，运行结果显示于下方的console中。（注：命令必须区分大小写，若输入大写Print则无法识别）



**四、配置 PyDev提高效率**

**4.1 显示源码行号**

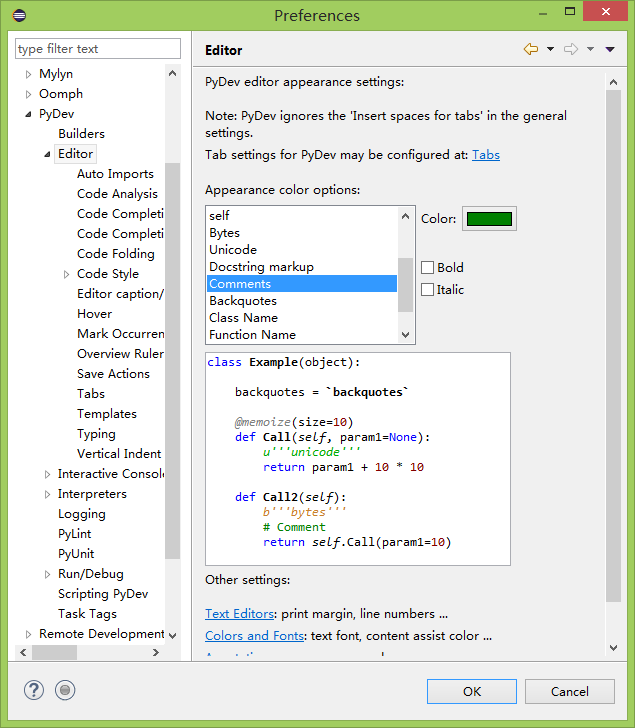
右键源码的左边缘，选中“Show Line Number



注：由于采用了QQ截图，右击时无法进行截图，故此处借用教程上的原图。

**4.2 修改注释颜色提高可读行**

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

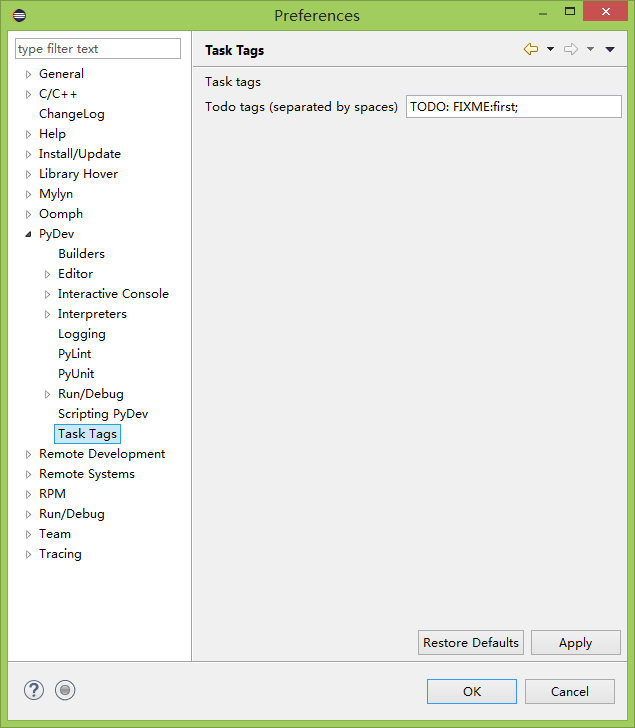


**4.3 配置任务标签**

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。

1. **配置任务标签：**

通过Window -> Preferences -> PyDev->Task Tags中配置任务标签:



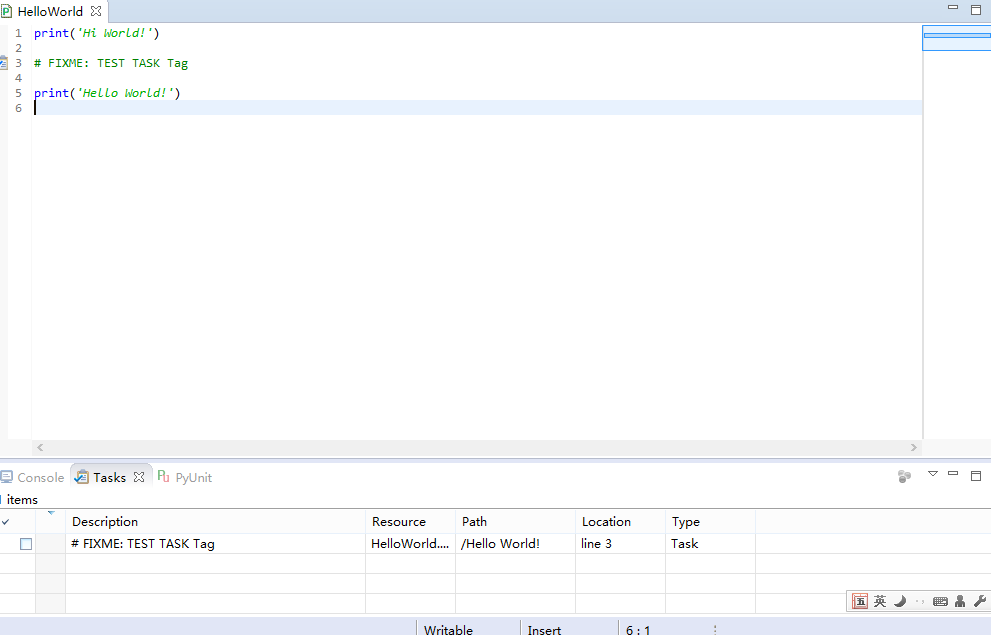
可以根据不同的任务类型，自己增加自定义标签，此处first为自定义标签。

**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：

# FIXME: TEST TASK Tag

并在该行左侧右键，选择Add Task



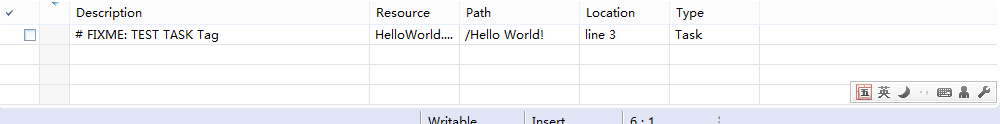
**3）识别任务**

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。



注：此处为借图，理由同4.1。

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。

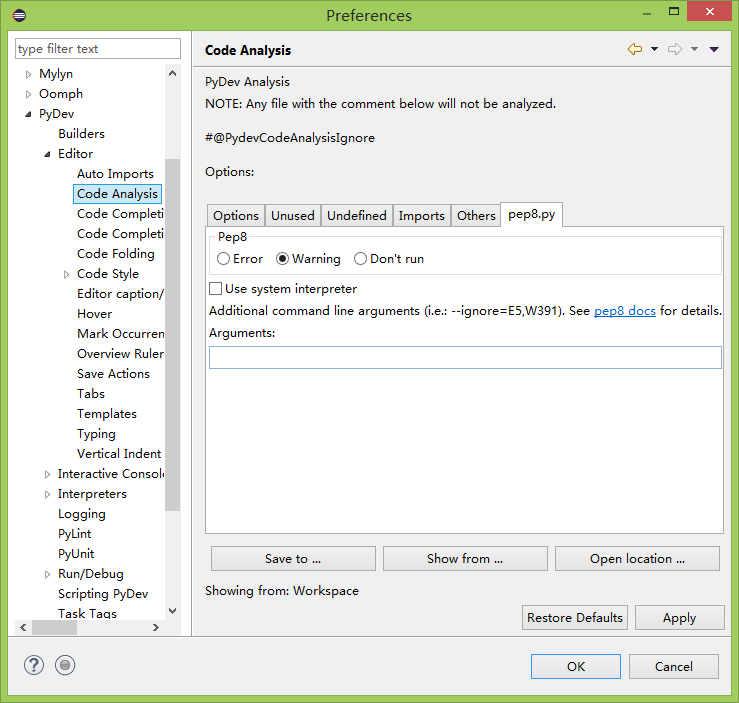
**5.1 PEP8检查和修改**

1）启动 pep8 检查：

Window > Preferences

PyDev > Editor > Code Analysis > pep8.py

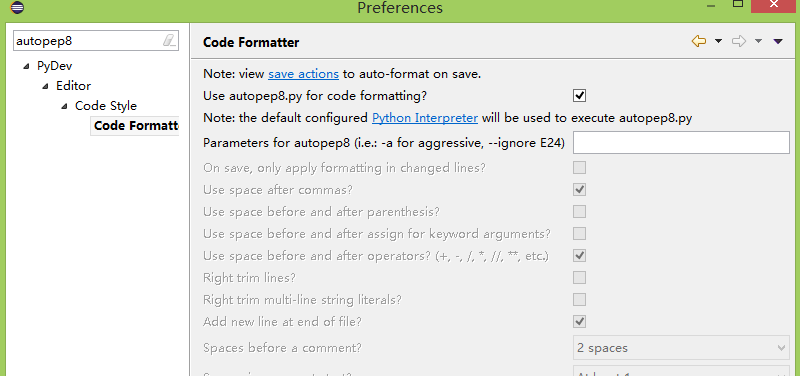
选择Errors/Warnings其中之一..



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?



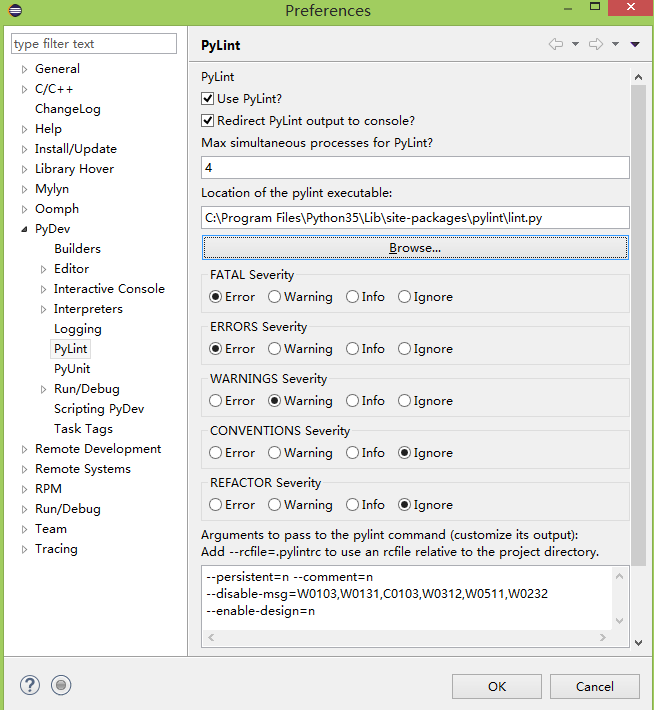
在Python源码窗口，按 CTRL-SHIFT-F 就可以自动修改代码。

**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,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。