

课程实践作业一：Python 学习和开发环境的建立

陆泽康

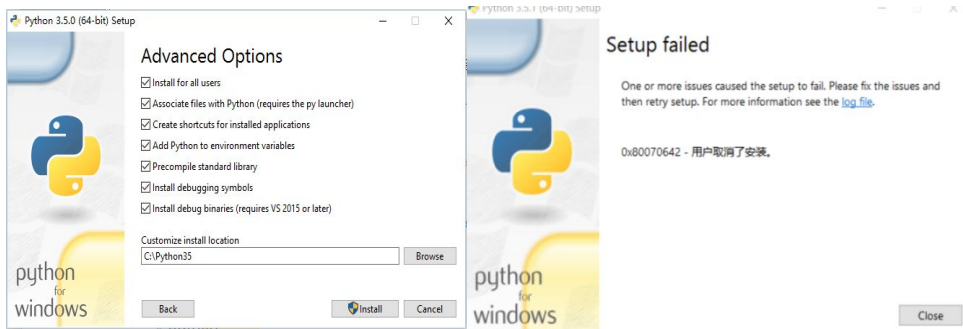
03014434

1. Python 基本环境的建立.....	3
2. Python 拓展包的安装.....	3
2.1 编码规范.....	3
2.2 科学计算机软件包.....	4
2.2.1 Scipy.....	4
2.2.2 交互计算机 jupyter.....	5
2.2.3 IF97 for Python.....	7
3. 基于 Eclipse 的开发环境.....	7
3.1 安装 JavaSDK.....	7
4. 安装 Python 开发插件 PyDev.....	9
5. 配置 PyDev 提高效率.....	10
5.1 显示源码行号.....	10
5.2 修改注释颜色提高可读行.....	11
6. 基于 PyDev 的语言规范静态检查.....	11
6.1 PEP8 检查和修改.....	11
6.2 使用 Pylint.....	12
小结.....	13
参考文献.....	13

安装 Python 程序，并根据需要安装了 Eclipse 以及配置 pyDev。记录其中的过程，完成指导手册。

1. Python 基本环境的建立

从老师那边拷贝 Python 程序的安装包，找到 python-3.5.1-amd64.exe, 点击安装。在过程中碰到错误，检查发现是网络连接的问题，问题解决。



安装结束之后，按下 Windows+x，找到输入命令符，输入

```
>pip install --upgrade pip
```

等待完成拓展包的安装，Python 安装初步结束。

2. Python 拓展包的安装

2.1 编码规范

找出命令提示符，输入 `pip install 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。)

输入 `pip install 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。)

```

C:\Users\Allen1>pip install autopep8
Collecting autopep8
  Downloading autopep8-1.2.2.tar.gz (105kB)
    100% |#####| 106kB 743kB/s
Collecting pep8>=1.5.7 (from autopep8)
  Downloading pep8-1.7.0-py2.py3-none-any.whl (41kB)
    100% |#####| 45kB 1.9MB/s
Installing collected packages: pep8, autopep8
Running setup.py install for autopep8 ... done
Successfully installed autopep8-1.2.2 pep8-1.7.0

C:\Users\Allen1>pip install pylint
Collecting pylint
  Downloading pylint-1.5.4-py2.py3-none-any.whl (547kB)
    100% |#####| 548kB 672kB/s
Collecting astroid<1.5.0,>=1.4.1 (from pylint)
  Downloading astroid-1.4.4-py2.py3-none-any.whl (211kB)
    100% |#####| 212kB 610kB/s
Collecting six (from pylint)
  Downloading six-1.10.0-py2.py3-none-any.whl
Collecting colorama (from pylint)
  Downloading colorama-0.3.7-py2.py3-none-any.whl
Collecting wrapt (from astroid<1.5.0,>=1.4.1->pylint)
  Downloading wrapt-1.10.6.tar.gz
Collecting lazy-object-proxy (from astroid<1.5.0,>=1.4.1->pylint)
  Downloading lazy_object_proxy-1.2.1-cp35-none-win_amd64.whl
Installing collected packages: wrapt, lazy-object-proxy, six, astroid, colorama, pylint
Running setup.py install for wrapt ... done
Successfully installed astroid-1.4.4 colorama-0.3.7 lazy-object-proxy-1.2.1 pylint-1.5.4 six-1.10.0 wrapt-1.10.6

```

2.2 科学计算机软件包

2.2.1 Scipy

输入地址：

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

从加州大学欧文分校下载编译好的 whl 文件：numpy scipy matplotlib

NuPy, a fundamental package needed for scientific computing with Pyt
 NumPy+MKL is linked to the latest Math Kernel Library and includes
 The vanilla builds are less tested and not compatible with binaries req
[numpy-1.10.4-mkl-cp27-cp27m-win32.whl](#)
[numpy-1.10.4-mkl-cp27-cp27m-win_amd64.whl](#)
[numpy-1.10.4-mkl-cp34-cp34m-win32.whl](#)
[numpy-1.10.4-mkl-cp34-cp34m-win_amd64.whl](#)
[numpy-1.10.4-mkl-cp35-cp35m-win32.whl](#)
[numpy-1.10.4-mkl-cp35-cp35m-win_amd64.whl](#)
[numpy-1.10.4-vanilla-cp27-cp27m-win32.whl](#)
[numpy-1.10.4-vanilla-cp27-cp27m-win_amd64.whl](#)
[numpy-1.10.4-vanilla-cp34-cp34m-win32.whl](#)
[numpy-1.10.4-vanilla-cp34-cp34m-win_amd64.whl](#)
[numpy-1.10.4-vanilla-cp35-cp35m-win32.whl](#)
[numpy-1.10.4-vanilla-cp35-cp35m-win_amd64.whl](#)
[numpy-1.11.0rc1-mkl-cp27-cp27m-win32.whl](#)
[numpy-1.11.0rc1-mkl-cp27-cp27m-win_amd64.whl](#)
[numpy-1.11.0rc1-mkl-cp34-cp34m-win32.whl](#)
[numpy-1.11.0rc1-mkl-cp34-cp34m-win_amd64.whl](#)
[numpy-1.11.0rc1-mkl-cp35-cp35m-win32.whl](#)
[numpy-1.11.0rc1-mkl-cp35-cp35m-win_amd64.whl](#)

SciPy is software for mathematics, science, and engineering.
 Requires numpy+mk1 and optionally pillow.
[scipy-0.17.0-cp27-cp27m-win32.whl](#)
[scipy-0.17.0-cp27-cp27m-win_amd64.whl](#)
[scipy-0.17.0-cp34-cp34m-win32.whl](#)
[scipy-0.17.0-cp34-cp34m-win_amd64.whl](#)
[scipy-0.17.0-cp35-cp35m-win32.whl](#)
[scipy-0.17.0-cp35-cp35m-win_amd64.whl](#)

Matplotlib, a 2D plotting library.
 Requires numpy, dateutil, pytz, pyparsing, cycler, setuptools
[matplotlib-1.5.1-cp27-cp27m-win32.whl](#)
[matplotlib-1.5.1-cp27-cp27m-win_amd64.whl](#)
[matplotlib-1.5.1-cp34-cp34m-win32.whl](#)
[matplotlib-1.5.1-cp34-cp34m-win_amd64.whl](#)
[matplotlib-1.5.1-cp35-cp35m-win32.whl](#)
[matplotlib-1.5.1-cp35-cp35m-win_amd64.whl](#)
[matplotlib-1.5.1.chm](#)
[matplotlib tests-1.5.1-py2.py3-none-any.whl](#)

下载好后输入 pip install wheel

```

C:\Users\Allen1>pip install wheel
Collecting wheel
  Downloading wheel-0.29.0-py2.py3-none-any.whl (66kB)
    100% |#####| 69kB 104kB/s
Installing collected packages: wheel
Successfully installed wheel-0.29.0

```

然后，用 pip 逐个本地安装，在命令行下输入

pip install *.whl (*为下载下来的文件名)

2.2.2 交互计算机 jupyter

1) 安装 Jupyter :

>pip install jupyter 在线安装

```
Collecting Jinja2 (from nbconvert->jupyter)
  Downloading Jinja2-2.8-py2.py3-none-any.whl (261kB)
    100% |#####| 266kB 1.3MB/s
Collecting ipython-genutils (from notebook->jupyter)
  Downloading ipython_genutils-0.1.0-py2.py3-none-any.whl
Collecting pyzmq<13 (from jupyter-client==4.1.1->qtconsole->jupyter)
  Downloading pyzmq-10.2.0-cp35-none-win_amd64.whl (767kB)
    100% |#####| 770kB 170kB/s
Collecting decorator (from traitlets->qtconsole->jupyter)
  Downloading decorator-4.0.9-py2.py3-none-any.whl
Collecting pickleshare (from ipython==4.0.0->ipykernel->jupyter)
  Downloading pickleshare-0.6-py2.py3-none-any.whl
Collecting setuptools>=18.5 (from ipython==4.0.0->ipykernel->jupyter)
  Downloading setuptools-39.2.0-py2.py3-none-any.whl (569kB)
    100% |#####| 512kB 629kB/s
Collecting simplegeneric>0.8 (from ipython==4.0.0->ipykernel->jupyter)
  Downloading simplegeneric-0.8.1.zip
Collecting jsonschema<2.5.0,>2.0 (from nbformat->nbconvert->jupyter)
  Downloading jsonschema-2.5.1-py2.py3-none-any.whl
Collecting MarkupSafe (from Jinja2->nbconvert->jupyter)
  Downloading MarkupSafe-0.23.tar.gz
Collecting path.py<6.2 (from pickleshare->ipython==4.0.0->ipykernel->jupyter)
  Downloading path.py-6.1.2-py2.py3-none-any.whl
Building wheels for collected packages: pyreadline, simplegeneric, MarkupSafe
  Running setup.py bdist_wheel for pyreadline ... done
  Stored in directory: C:\Users\Allen\AppData\Local\pip\Cache\wheels\55\7d\c4\56ba8f63e20ef39b55f1f938076ac4810d5193
  Running setup.py bdist_wheel for simplegeneric ... done
  Stored in directory: C:\Users\Allen\AppData\Local\pip\Cache\wheels\81\3c\13\2b621669f3ba74d01386a5ec6a0f0f724848ee2
  Running setup.py bdist_wheel for MarkupSafe ... done
  Stored in directory: C:\Users\Allen\AppData\Local\pip\Cache\wheels\94\c4\79\c79a998b64e1281c699a96bd33cfe8218d17a7
Successfully built pyreadline simplegeneric MarkupSafe
Installing collected packages: ipython-genutils, decorator, traitlets, jupyter-core, pyzmq, jupyter-client, to
-py, pickleshare, setuptools, simplegeneric, ipython, ipykernel, pygments, qtconsole, jsonschema, nbformat, s
MarkupSafe, Jinja2, nbconvert, notebook, ipynbdata, pyreadline, jupyter-console, jupyter
Found existing installation: setuptools 18.2
Uninstalling setuptools-18.2:
  Successfully uninstalled setuptools-18.2
```

2) 安装 Python 语言内核

>pip install ipython , 支持Python 语言

```
C:\Users\Allen1>pip install ipython
Requirement already satisfied (use --upgrade to upgrade): ipython in c:\python35\lib\site-package
Requirement already satisfied (use --upgrade to upgrade): decorator in c:\python35\lib\site-packa
Requirement already satisfied (use --upgrade to upgrade): simplegeneric>0.8 in c:\python35\lib\si
on)
Requirement already satisfied (use --upgrade to upgrade): traitlets in c:\python35\lib\site-packa
Requirement already satisfied (use --upgrade to upgrade): pickleshare in c:\python35\lib\site-pac
Requirement already satisfied (use --upgrade to upgrade): setuptools>=18.5 in c:\python35\lib\sit
n)
Requirement already satisfied (use --upgrade to upgrade): ipython-genutils in c:\python35\lib\sit
ets->ipython)
Requirement already satisfied (use --upgrade to upgrade): path.py>=6.2 in c:\python35\lib\site-pa
e->ipython)
C:\Users\Allen1>
```

3) 安装依赖包

>pip install pyreadline

>pip install sympy

```

C:\Users\Allen1>pip install sympy
Collecting sympy
  Downloading sympy-1.0.tar.gz (4.3MB)
    100% |#####| 4.3MB 88kB/s
Collecting mpmath>=0.19 (from sympy)
  Downloading mpmath-0.19.tar.gz (498kB)
    100% |#####| 499kB 160kB/s
Building wheels for collected packages: sympy, mpmath
  Running setup.py bdist_wheel for sympy ... done
  Stored in directory: C:\Users\Allen1\AppData\Local\pip\Cache\wheels\1e\9d\5b\b16fed9678e60b8deed
  Running setup.py bdist_wheel for mpmath ... done
  Stored in directory: C:\Users\Allen1\AppData\Local\pip\Cache\wheels\9c\ed\bc\b857365dc81856c4b200
Successfully built sympy mpmath
Installing collected packages: mpmath, sympy
Successfully installed mpmath-0.19 sympy-1.0

```

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()
```

```

C:\Users\Allen1>IPython
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 b
Type "copyright", "credits" or "license" for more information.

IPython 4.1.2 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]: from IPython.external.mathjax import install_mathjax install_mathja
File "<ipython-input-1-6782bf366ea8>", line 1
      from IPython.external.mathjax import install_mathjax install_mathjax
SyntaxError: invalid syntax

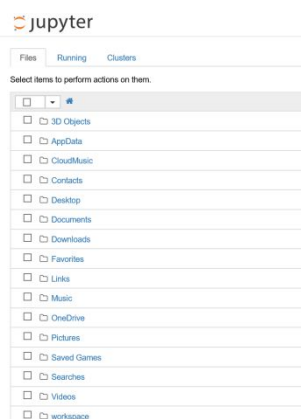
In [2]:

```

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



3. 基于 Eclipse 的开发环境

3.1 安装 JavaSDK

找到拷贝文件中的 jdk-8u74-windows-x64，根据实际配置安装。



Eclipse IDE 是使用 Java 开发的，电脑中安装好 Java JRE/JDK 软件包，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，如下图



```
命令提示符
Microsoft Windows [版本 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Allen1>java -version
java version "1.8.0_74"
Java(TM) SE Runtime Environment (build 1.8.0_74-b02)
Java HotSpot(TM) 64-Bit Server VM (build 25.74-b02, mixed mode)

C:\Users\Allen1>
```

3.2 Eclipse IDE 的安装

首先打开 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）

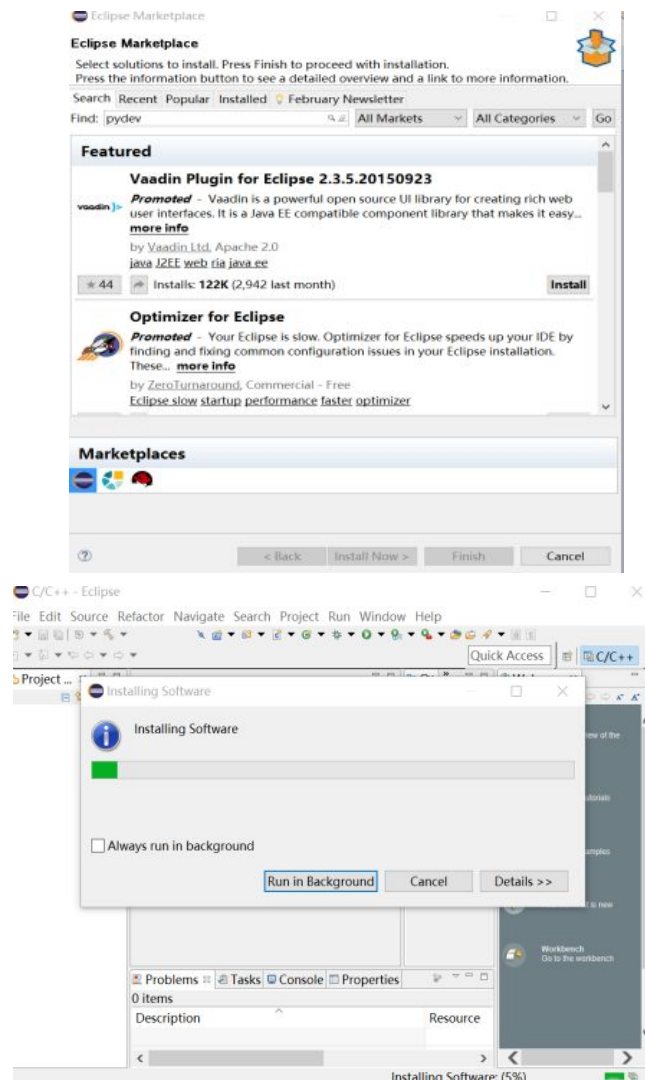


configuration	2016/3/8 14:56	文件夹	
dropins	2015/9/24 6:26	文件夹	
features	2016/3/8 14:50	文件夹	
p2	2016/3/8 14:56	文件夹	
plugins	2016/3/8 14:51	文件夹	
readme	2016/3/8 14:35	文件夹	
.eclipseproduct	2015/9/2 10:05	ECLIPSEPRODUCT ...	
artifacts	2016/3/8 14:56	XML 文档	14
eclipse	2015/9/24 6:28	应用程序	31
eclipse	2016/3/8 14:56	配置设置	
eclipsec	2015/9/24 6:28	应用程序	2
notice	2015/8/17 11:43	HTML 文件	

4. 安装 Python 开发插件 PyDev

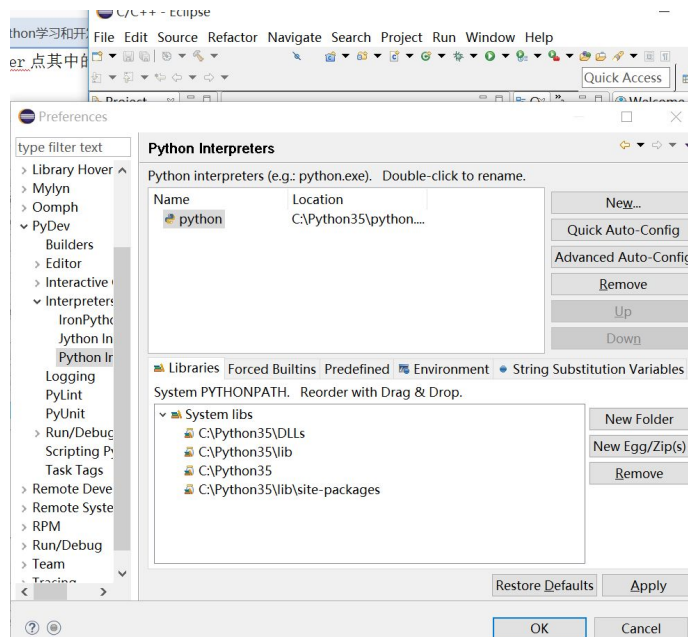
4.1 安装 PyDev 插件

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，安装结束点击重新启动。



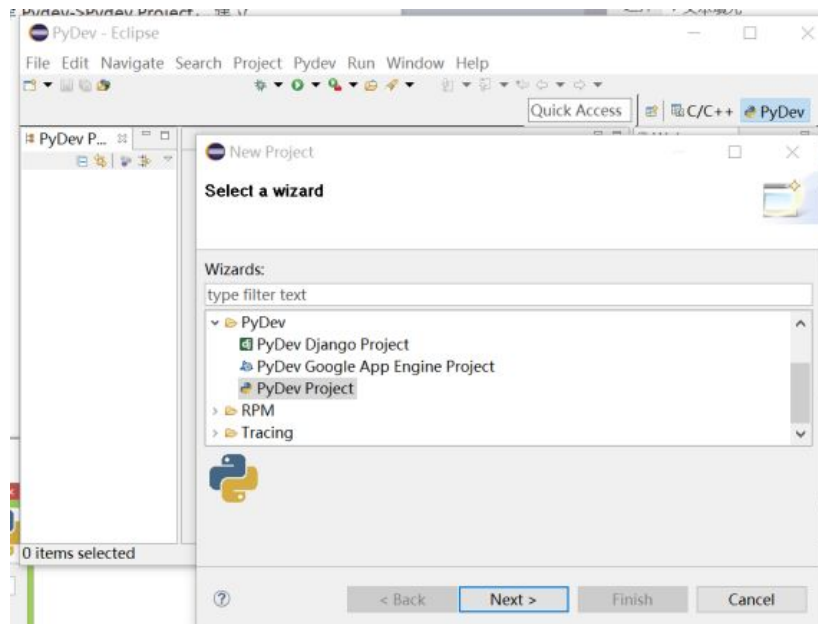
Windows->Preference->Pydev->Interpreters->Python Interpreter 点其中的：Advanced

Auto-config 配置开发使用的 Python 解释器版本，配置 Python 解释器。切换到 Python 场景就能开始开发了。



5. 配置 PyDev 提高效率

5.1 新建一个软件工程 Hello world

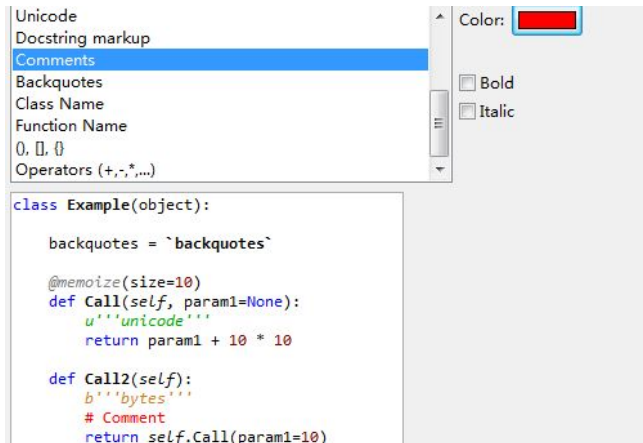


5.1 显示源码行号

右键源码的左边缘，选中 Show Line Number（已经默认选中）。

5.2 修改注释颜色提高可读性

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



6.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。

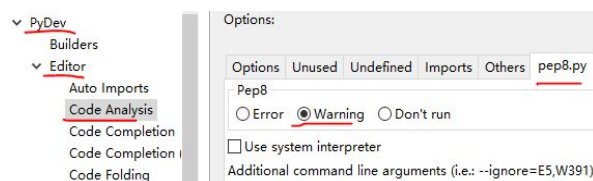
6.1 PEP8 检查和修改

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: .

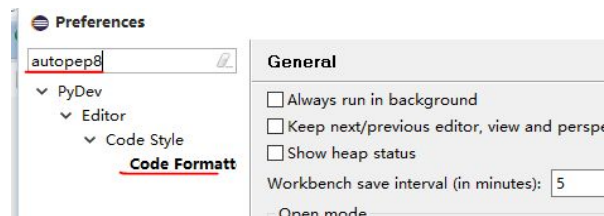
```

1  # -*- coding: utf-8 -*-
2
3  import TagDefToRedisHash as tagdef
4
5  if __name__ == "__main__":
6
7      rowbeginindex=2
8      colidindex=2
9      coldescindex=1
10
11     excelfile = tagdef.open_excel(u'./cs_tag_a
12
13     tagdeflist =tagdef.tagdef_from_excel_sheet
14
15     tagdef.TagDefToRedisHashKey(tagdeflist)
16

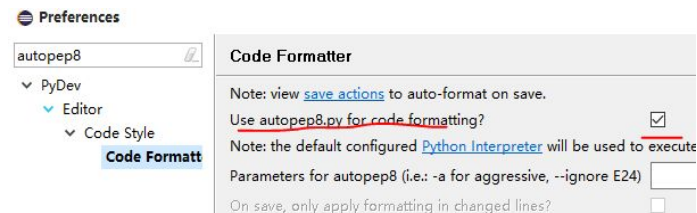
```

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

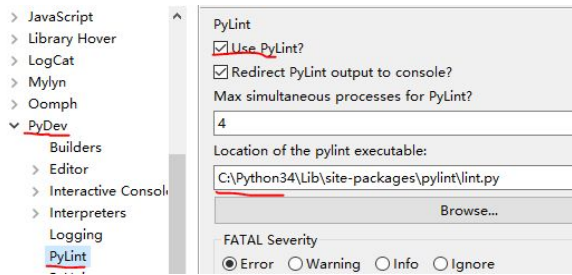
6.2 使用 Pylint

PyDev 默认不开启 Pylint。通过

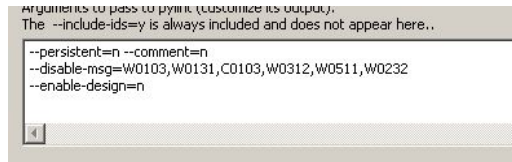
Window -> preferences -> Pydev -> Pylint, 选中 "Use pylint?",

找到安装好的 lint.py 的地址, 例如

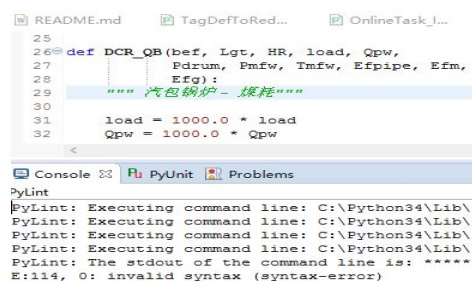
"C:\Python34\Lib\site-packages\pylint\lint.py"



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

本指导手册主要介绍了, 首先是 IDLE, 其他的还有 Eclipse 加 PyDev。过程中感受到了 Python 开发环境的多种多样。本次只有这几个开发环境, 以后会根据自已的需求安装更多的开发环境。

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