软件工程基础及实践

作业二——建立Python基本开发环境

东南大学 能源与环境学院

03014321 于天池

**目录**

[1 Python基本介绍及开发环境的建立 1](#_Toc477620367)

[1.1 Python3.5.0安装过程 1](#_Toc477620368)

[1.2 Python3.5.0使用示例——IDLE开发环境 3](#_Toc477620369)

[2 扩展模块 4](#_Toc477620370)

[2.1 安装科学计算包 4](#_Toc477620371)

[2.2 规范格式 6](#_Toc477620372)

[2.3 IF97——水和水蒸气物性计算 6](#_Toc477620373)

[2.4语言计算包 7](#_Toc477620374)

[3 建立交互计算环境Jupyter Notebook 7](#_Toc477620375)

[3.1 Jupyter Notebook安装过程 7](#_Toc477620376)

[3.2 Jupyter Notebook使用示例 9](#_Toc477620377)

[4 集成开发环境Eclipse 11](#_Toc477620378)

[4.1 Eclipse安装过程 11](#_Toc477620379)

[4.1.1 安装Java SDK并配置环境变量 11](#_Toc477620380)

[4.1.2 安装Eclipse及配置工作空间 12](#_Toc477620381)

[4.1.3 PyDev插件的安装与解释器版本配置 13](#_Toc477620382)

[4.1.4其他配置 16](#_Toc477620383)

[4.1.5语言规范静态检查 19](#_Toc477620384)

[4.2 Eclipse配置——文件字符编码方式 21](#_Toc477620385)

[4.2.1 工作空间编码方式 21](#_Toc477620386)

[4.2.2 工程编码方式 22](#_Toc477620387)

[4.3 Eclipse使用示例 22](#_Toc477620388)

[5 其他问题 24](#_Toc477620389)

[6 个人小结 25](#_Toc477620390)

[7 参考资料 26](#_Toc477620391)

# 1 Python基本介绍及开发环境的建立

Python是一种解释型、面向对象、动态数据类型的高级程序设计语言，是纯粹的自由软件，具有丰富和强大的库。它之前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。Python语言的风格简洁而清晰，有大量适用性甚广的类库和python开源框架可以使用，现在Python已经从最开始的原型开发发展成为大数据的主流开发语言，适用于各种操作系统，目前在国际上非常流行，正在得到越来越多的应用。

Python的开发环境十分友好，下面我们开始搭建Python的基本开发环境。

## 1.1 Python3.5.0安装过程

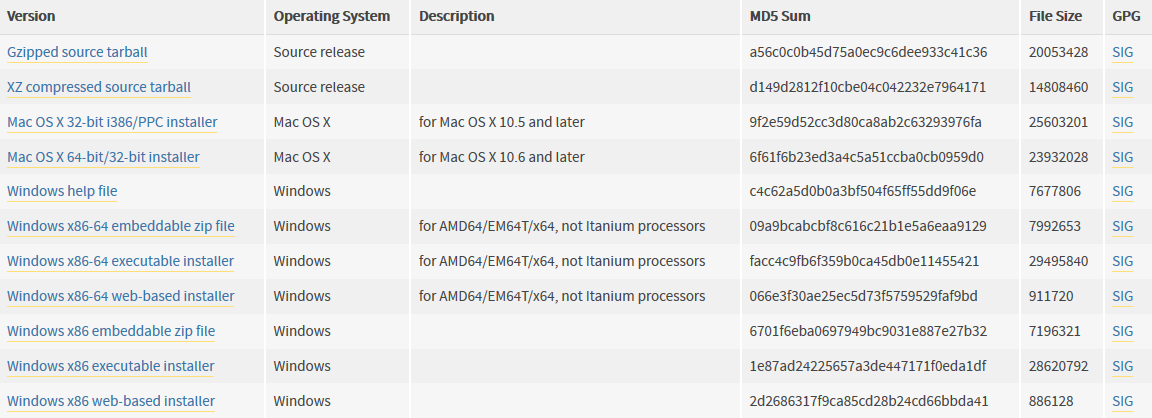
首先我们进入Python官网（<https://www.python.org/>）的download区，为了适配我个人电脑的配置（64位的Windows 7的操作系统），选用较老的版本Python3.5.0，选择[Windows x86-64 executable installer](https://www.python.org/ftp/python/3.5.0/python-3.5.0-amd64.exe)版本进行下载安装。

图1 Python3.5.0下载区

开始安装后，需要做以下几步：

①选中Add Python 3.5 to PATH（原因是Windows需要根据Path的环境变量设定的路径查找python.exe）；

②点击Customize installation，除最后一个选项外，其他全部选择（最后一个选项是指可以借助VS2015进行调试，我们暂时不需要此项）；

③建议定义安装文件夹，安装至C:/Python35 。

具体如下图所示：

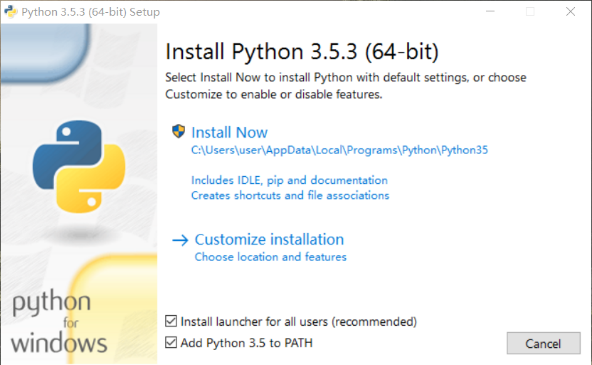
****

图2 选中Add Python 3.5 to PATH

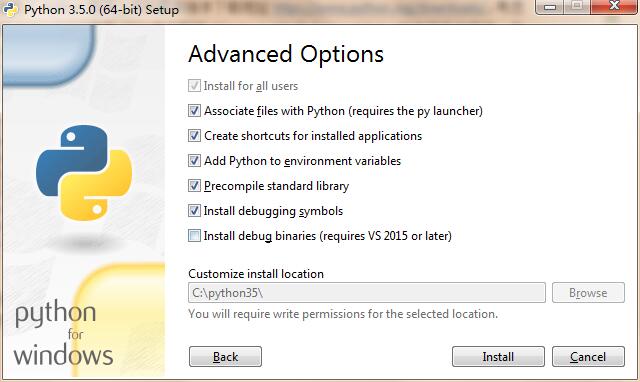


图3 Advanced Options

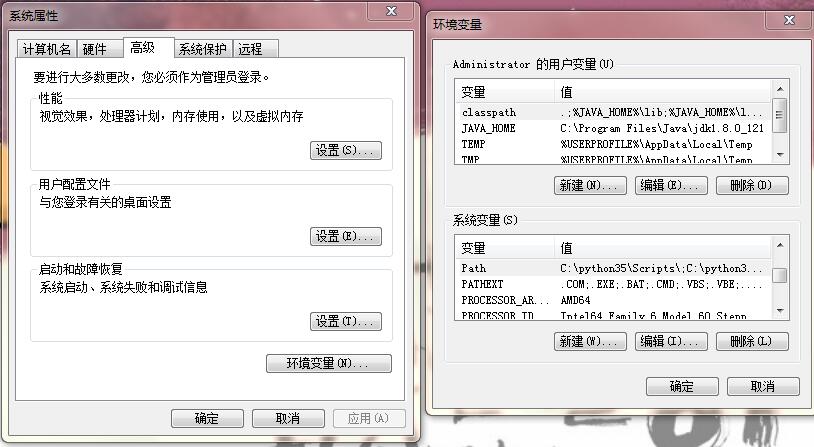
**特别说明**：如果在安装时忘记勾选Add Python 3.5 to PATH，需要将python.exe所在的路径添加到环境变量的Path中去，如图所示：

图4 手动添加变量

安装成功后，打开命令提示符，在DOS命令窗口中输入 >python –m pip install –U pip指令，检查Python是否安装成功，并更新pip至最新版本。

图5 命令提示符输入指令后显示pip已经最新

至此，pip已经更新完成，Python3.5.0安装成功。

## 1.2 Python3.5.0使用示例——IDLE开发环境

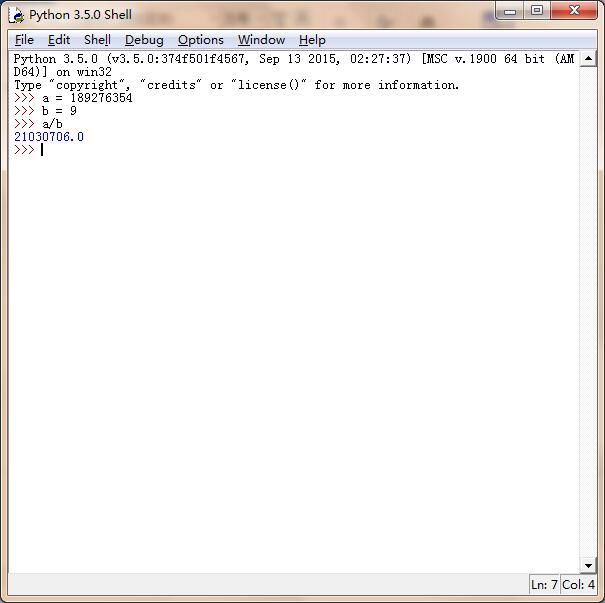
IDLE是Python的集成开发环境，此时IDLE已经建立，可以进行简单的程序开发示例。 

图6 开发环境IDLE

**注**：除了IDLE可以对Python进行程序开发外，Python还有其他两种运行方式：

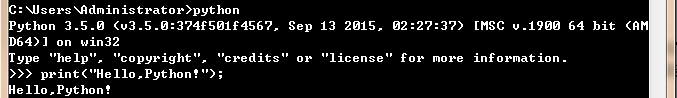
1. **Python命令行**——在DOS窗口输入>python后就会进入python shell中了，输入一段代码，回车，会立即显示出结果，是一个很方便的学习Python语言的途径。

图7 python shell

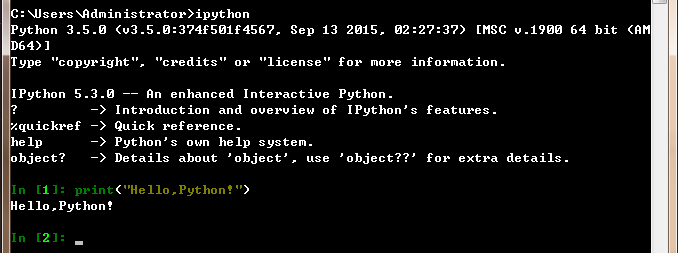
1. **交互式编程**——不需要创建脚本文件，通过Python解释器的交互模式进来编写代码，进入方式是在命令行中输入ipython，回车即可。ipython 是一个Python 的交互式Shell，比默认的Python Shell好用得多，功能也更强大。她支持语法高亮、自动完成、代码调试、对象自省，支持 Bash Shell 命令，内置了许多 很有用的功能和函式等，非常容易使用。

图8 交互式编程

1. 运行Python的脚本文件——通过DOS命令运行脚本文件或者直接双击 .py文件运行。

# 2 扩展模块

## 2.1 安装科学计算包

为了丰富其功能，我们需要下载安装一些拓展模块，其中基本的科学计算包有 SciPy、NumPy、Matplotlib。

**下载途径：**加州大学欧文分校的网站（<http://www.lfd.uci.edu/~gohlke/pythonlibs/>）。

**安装方式：**找到对应版本的安装包，下载后用命令提示符进行pip安装：

>pip install {filename}.whl

**安装科学计算包示例：**SciPy

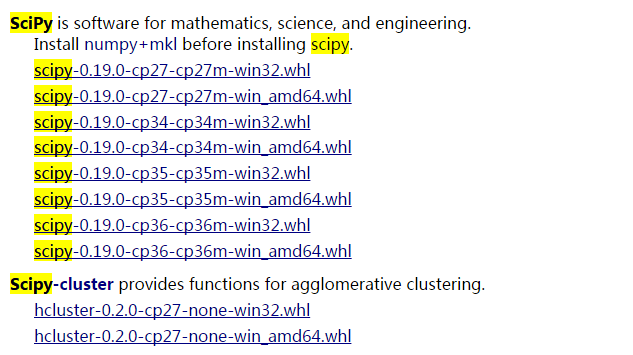
进入欧文分校网站，找到Scipy版本区域，根据笔者电脑配置，选用scipy-0.19-cp35-cp35m-win\_amd64.whl进行下载：

图9 SciPy扩展包下载选项

另外，该网站上提供的文件十分丰富，为了在众多的文件中快速找到所需要的拓展包，我们可以进行本网页查找功能（笔者采用的是谷歌浏览器）：

图10 SciPy扩展包查找

**安装过程：**打开命令提示符，输入>pip install scipy-0.19.0rc2-cp35-cp35m-win\_amd64.whl，由于笔者之前已经提前安装完成，下图显示为已经安装完毕。

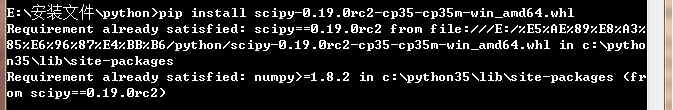


图11 Scipy科学扩展包安装成功

**特别说明：**

1. 需要从现在的目录下cd到安装包所对应的下载的目录下才能找到SciPy文件并且安装，cd方式如下：

图12 cd到对应目录

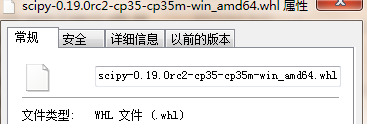
1. 为了保证安装正确，可以进入下载的安装文件的属性，将文件名复制。

图13 属性

1. 根据提示，需要先安装numpy+mkl，然后才能安装SciPy，此处只是作为演示。

## 2.2 规范格式

依次安装完毕上述科学计算包后，仍需要安装扩展模块pep8、autopep8和pylint。

pep8：Python增强建议书；

autopep8： Python代码自动排版工具；

pylint：代码检查工具；

**安装方式：**安装扩展模块pep8、autopep8和pylint时，不需要下载，直接在DOS窗口中输入>pip install pep8（autopep8或pylint）即可在线安装。

## 2.3 IF97——水和水蒸气物性计算

IF97是水和水蒸气物性计算的语言包，也是为了丰富我们在Python编程过程中的功能。我们可以在GitHhb的IF97共享库及其Python封装包仓库中下载zip文件：

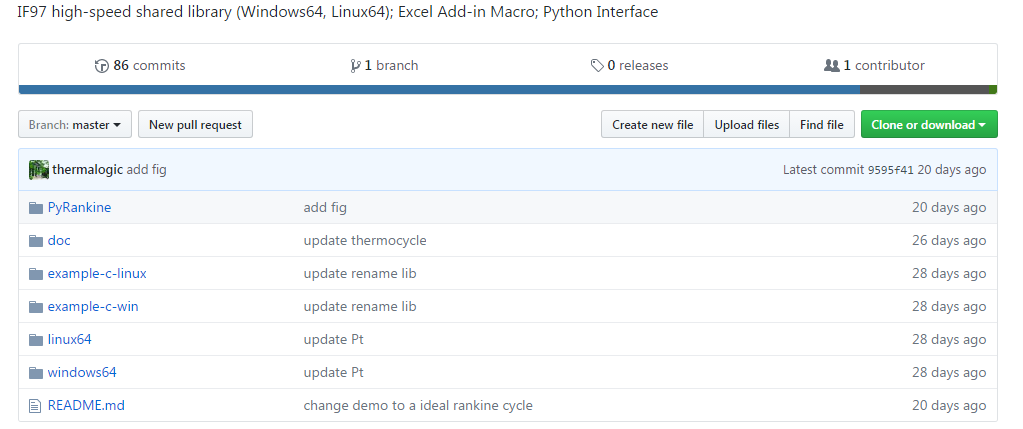
<https://github.com/PySEE/SEUIF97>

图14 下载SEUIF97zip文件

下载后，解压zip文件，并将和操作系统对应版本（window64）的libseuif97.dll拷贝到c:\windows\system，然后把seuif97.py拷贝到python安装目录的lib子目录下(c:\python35\Lib)，完成。

## 2.4语言计算包

**安装方式：**在线安装，DOS窗口中输入>pip install iapws

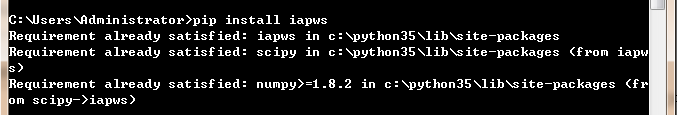
****

图15 语言计算包安装成功

至此，拓展模块的安装告一段落。我们仍然可以通过类似的方式，为Python的开发环境添加更多的功能模块，大家可以根据自己的需要和兴趣进行自由补充。

# 3 建立交互计算环境Jupyter Notebook

## 3.1 Jupyter Notebook安装过程

Python3.5.0的基本开发环境建立成功后，我们将进行Jupyter Notebook开发环境的建立和安装。

**安装方式：**在线安装，在DOS命令行中直接输入命令>pip install jupyter即可。

由于某些安装包在线时间长且难以下载，可以选择从加州大学欧文分校下载相应的软件包的whl，然后在在DOS窗口中pip安装。笔者选用了直接在线安装：

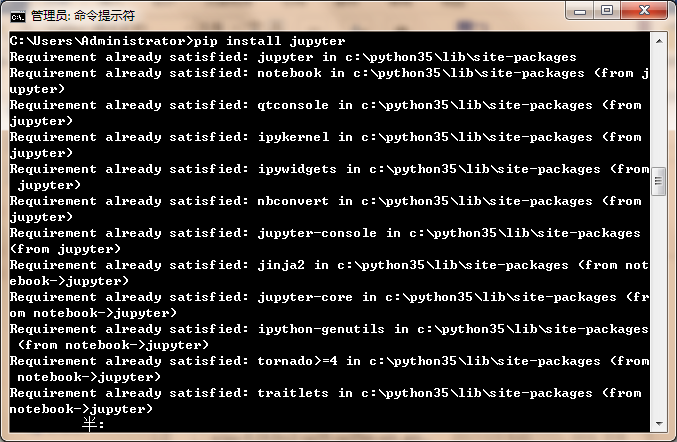


图16 已安装Jupyter Notebook

安装成功后，在命令提示符中的制定目录下，输入>jupyter notebook ，可以直接运行并用notebook模式打开这一目录，弹出的浏览器界面如下图：

图17 Jupyter Notebook网页界面

因此我们可以了解到，可以通过Jupyter Notebook在浏览器的网页上将本机的相应目录的文件复现，并且我们可以在网页上进行操作。因此我们可以这样理解，Jupyter Notebook类似于草稿纸，可以在其上面进行思考探索过程的演示，而且可以看到结果，实际上它就是一个在线编辑器。

## 3.2 Jupyter Notebook使用示例

正如之前所说，我们可以通过Jupyter Notebook 在线对复现的文件夹的内容进行编译。

首先我们为了方便进入任何目录下的网页，而不仅仅是默认目录的复制，可以进行以下工作：

**方法一：**在打开Jupyter之前，先采用cd转换DOS指向地址，具体方法可参考2.1章节中提到的方法。改变默认地址后，再次输入>jupyter notebook，即可实现任何目录的文件复制。

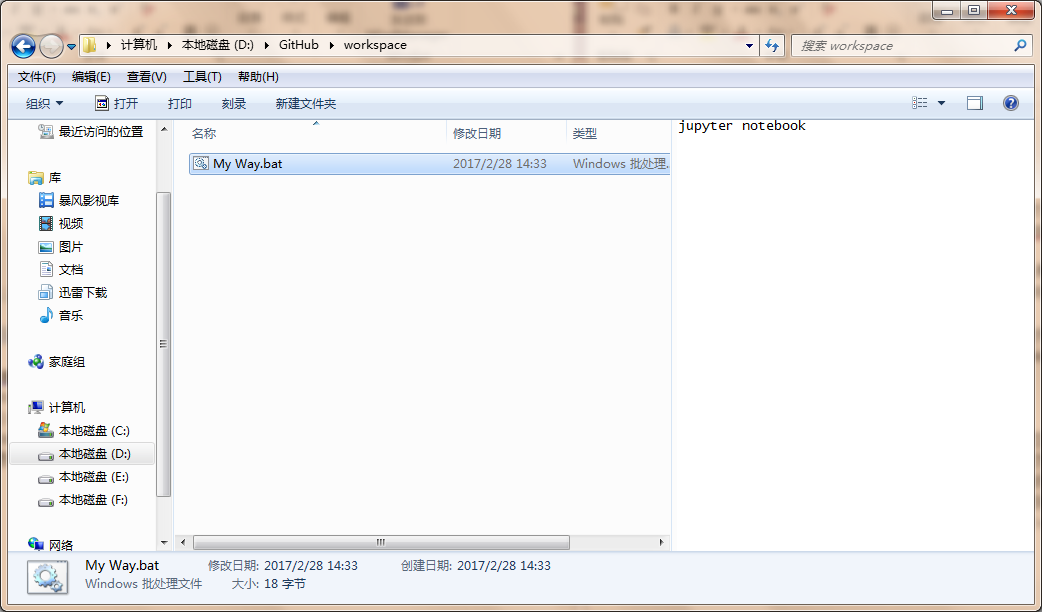
**方法二：**快速通道——直接建立一个扩展名为.bat的文本文件，命名为My Way.bat，并在文件中输入命令：jupyter notebook，即可实现双击该文件立即打开对应文件夹的网页。

图18 示例

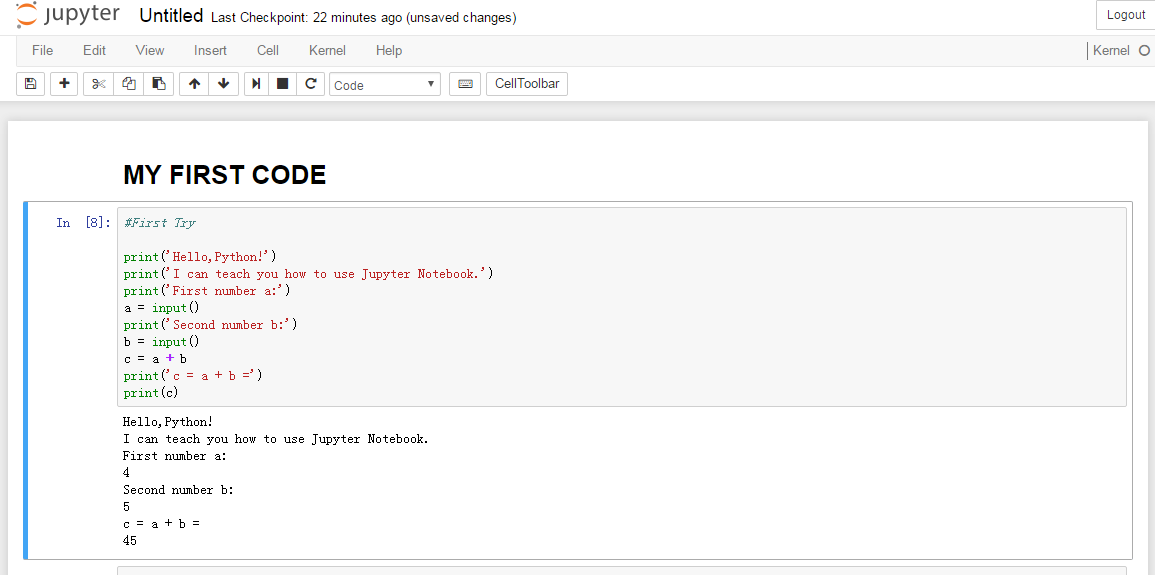
因此，我们可以了解到，我们完全可以创建一个自己的工作空间。在这个新建的工作目录下，可以新建自己的文件，在文件中进行编码，我们所进行代码编写的一行行可以输入的窗口即称为“cell”。

图19 Jupyter Notebook使用示例截图

其中的功能十分强大，除了直接编写code，也写入其他的格式，既可以有markdown的代码，也可以有python的代码，皆可以独自编译；我们可以进行代码的复制、换行，编写标题，甚至可以在这个文件中导入其他的代码。总之，有很多可以优化操作的功能，需要我们去具体探讨和学习。

**特别说明：**为了方便学习，我们可以将上课所需的课程文件拷贝到PC上然后再用Jupyter Notebook打开。完成拷贝后，找到notebook 下的StartNB.bat文件，双击运行，即可用Jupyter Notebook打开对应的网页。当然也可以用之前提过的其他方法打开。

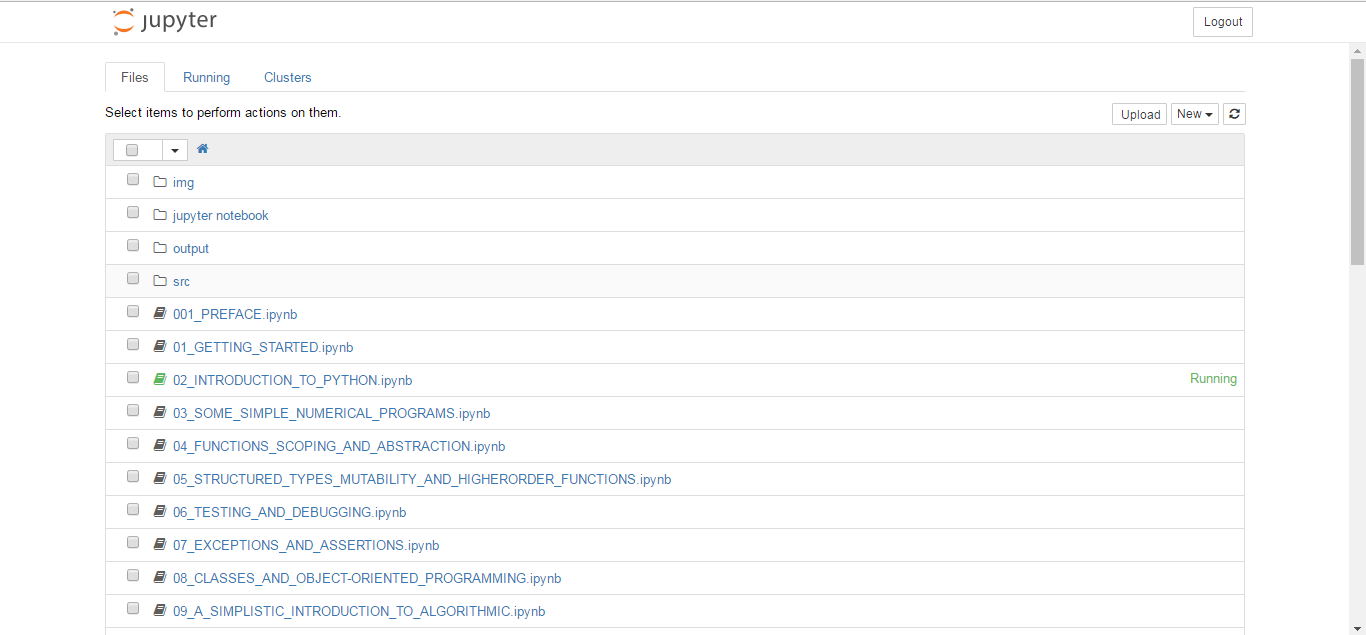


图20 Jupyter Notebook运行课件目录的界面截图

# 4 集成开发环境Eclipse

Eclipse 是一个开放源代码的、基于Java的可扩展开发平台，可以说它是一个框架和一组服务，用于通过插件组件构建Python的语言开发环境。

## 4.1 Eclipse安装过程

### 4.1.1 安装Java SDK并配置环境变量

Eclipse IDE是使用Java开发的，我们需要预先安装好Java JRE/JDK软件包。我们可以通过在DOS命令窗口中输入：>java –version来检查本机是否已经安装了Java软件包。

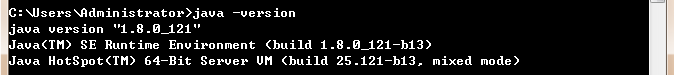
如果电脑中已经安装了Java ，会显示有关版本信息。

图21 已安装最新版本的java

如果没有安装Java或者现有版本不能满足Eclipse IDE最低要求，需要从Oracle下载Java包进行安装。

**下载地址：**

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

**安装方式：**进入网址后，根据PC情况，选择对应系统的Java进行下载。

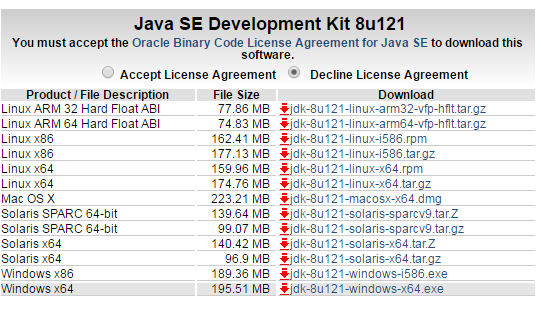
Windows 32位：\*-windows-i586.exe；Windows 64位：\*-windows-i64.exe.。

图22 Java下载

在Java安装完成后，必须注意的是，要对环境变量进行配置。

**配置方法：**配置java环境变量（类似于1.1章节最后提到的方法），从控制面板中进入电脑的高级系统设置，进入环境变量，在系统环境变量中配置具体的环境变量。新建一个变量，命名为“classpath”，值为“.;%JAVA\_HOME%\lib;%JAVA\_HOME%\lib\tools.jar”。同样，新建名为”JAVA\_HOME“的变量名，变量值为之前安装jdk的目录。

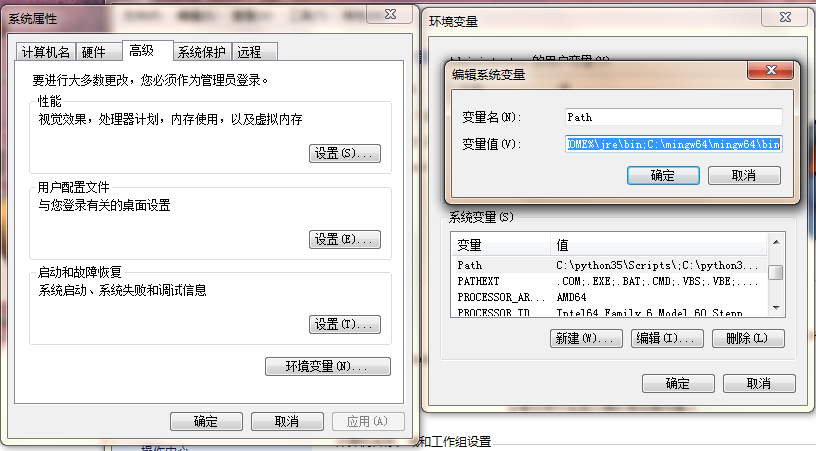
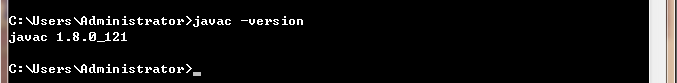
最后，找到系统变量path的变量值，加上 “;%JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin”，自此java的配置完成。

图23 配置环境变量

最终在DOS中输入>javac -version检验是否配置成功。

图24 检验java配置成功

### 4.1.2 安装Eclipse及配置工作空间

Eclipse IDE是一种集成开发环境，有很多版本。考虑到本课程使用Python语言，之后也可能会涉及C/C++开发，所以下载Eclipse CDT(Eclipse IDE for C/C++)版。

**Eclipse CDT官方下载地址：**

<http://www.eclipse.org/downloads/eclipse-packages/>

根据个人PC的情况下载相应版本的文件，下载完成后，解压到指定目录，运行eclipse.exe即可。安装过程比较简单，不再赘述。

**工作空间配置：**首次打开Eclipse软件时，软件会提示设置默认的工作空间。为了方便大家操作，最好在非系统盘中，例如笔者在D盘建立一个字目录，以后所有开发的项目都将默认保存到该目录下。操作方法是当其提示设置默认的工作空间时，“Browser”到该目录，勾选“Use this as the default and do not ask again",就将该目录配置为Eclipse默认当前工作空间目录了。

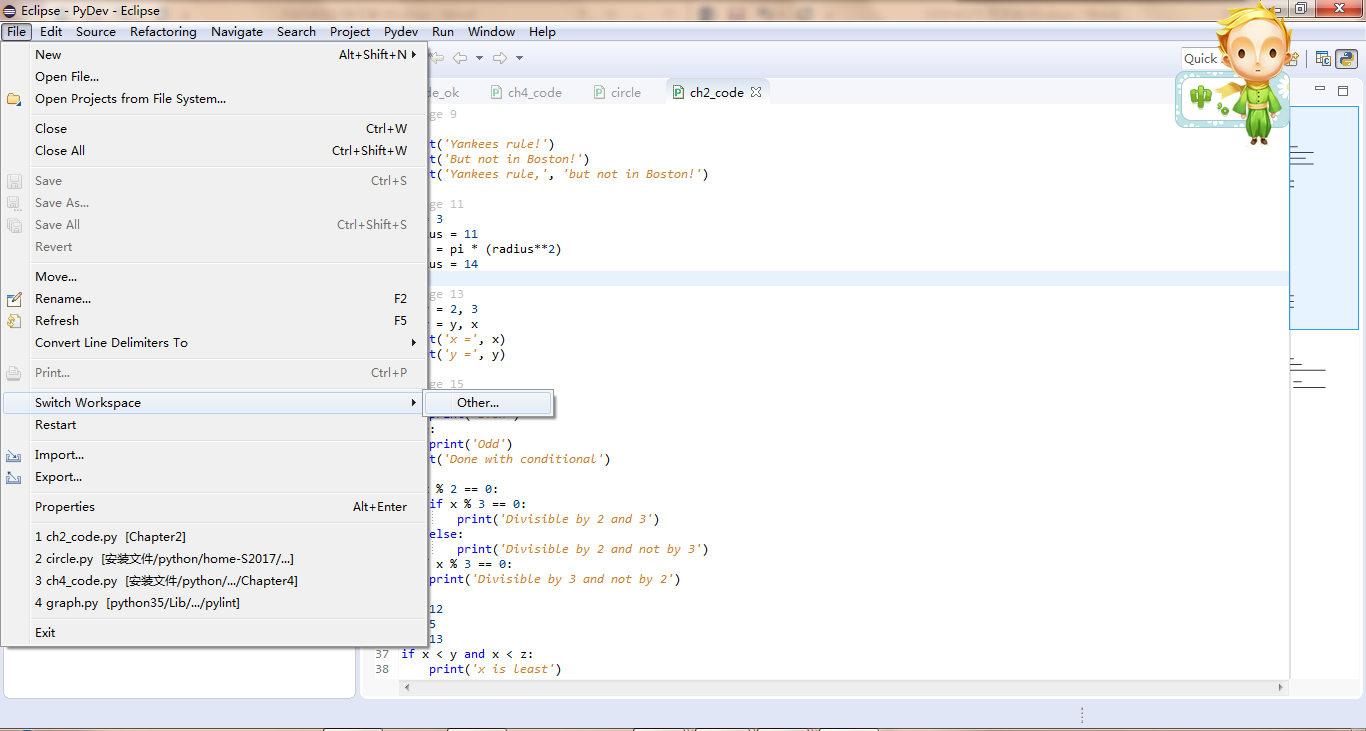
**特别说明：**可能大家不小心错过最初的修改目录，或者想建立多个工作空间目录，可以通过“File->Switch Workspace->other”改变当前工作空间目录进行重新配置。

图25 更改默认工作目录

### 4.1.3 PyDev插件的安装与解释器版本配置

为了更好的实现Eclipse平台的功能，也为了使操作者更加得心应手，我们需要进行PyDev的安装配置。

**安装方式：**进入Eclipse界面，通过Help->Eclipse Marketplaces进入市场，查找pydev项目，选择需要的插件，点击“install”在线安装（当然也可以从PyDev官网上(<http://www.pydev.org/>)进行直接下载）。

图26 安装Pydev

**配置Python解释器：**

安装好后，重启Eclipse，通过Windows->Preference->Pydev->Interperters->Python Interperter，点击Advanced Auto-config，配置开发使用的 Python解释器版本。

配置好后，为方便之后的操作，可以添加Python场景到工作页面。

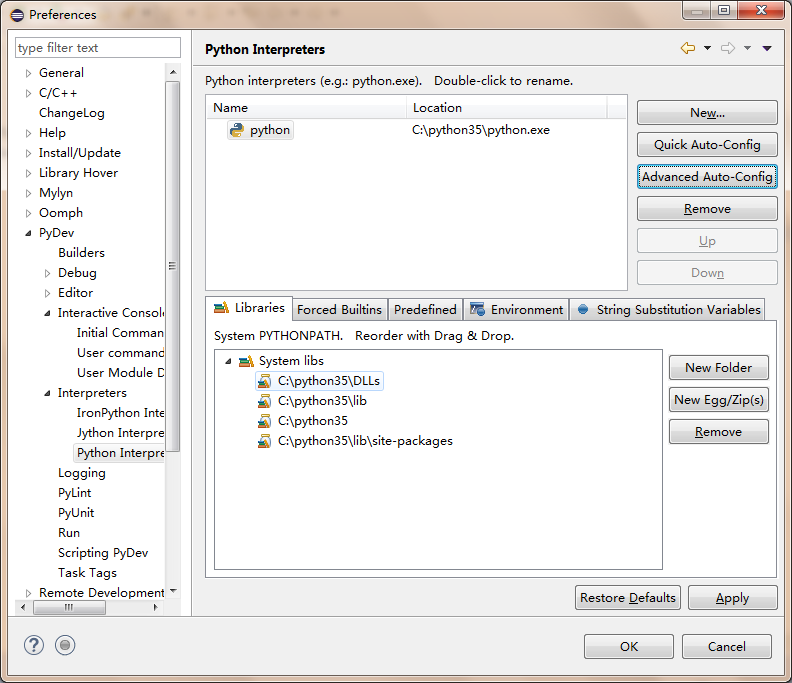
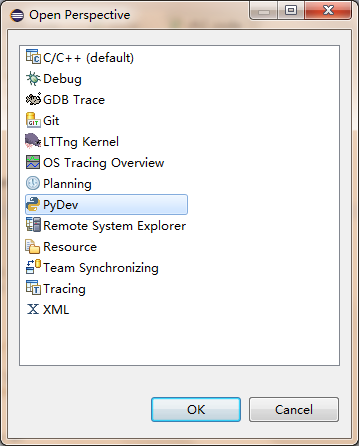
图27 配置解释器版本

图28 添加Python场景到工作页面

### 4.1.4其他配置

为了方便使用，还可以对Pydev进行其他配置。

1. **修改编辑器颜色提高可读性**

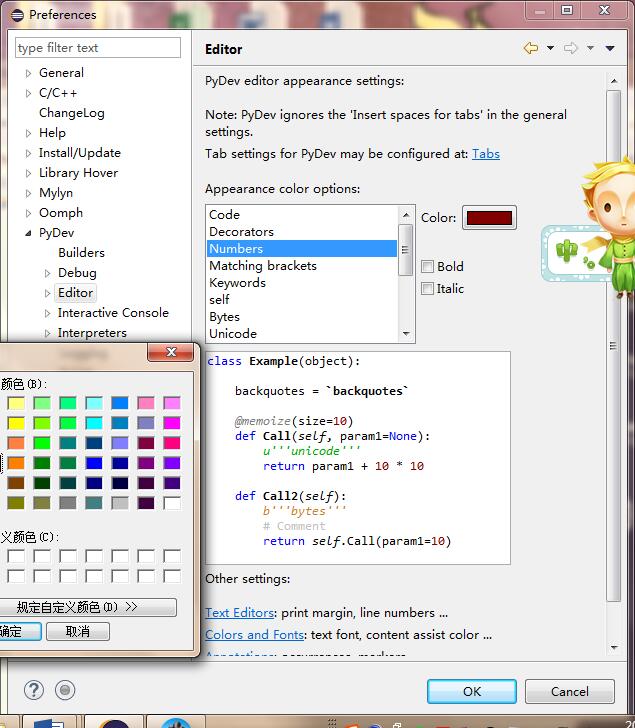
从Window → Preferences→PyDev→Editor,进入配置界面：

图29 修改编辑器配色

配置完成后，在任何.py文件中编写的程度都同预览中的配色一致。

1. **添加任务标签注释**

“任务标签”在代码中可以对任务进行标识，在开发环境中进行识别，并加入工作空间的任务列表中。我们可以通过开发程序过程中使用“任务标签”，并在代码中识别当前任务状态。

1. **配置任务标签**

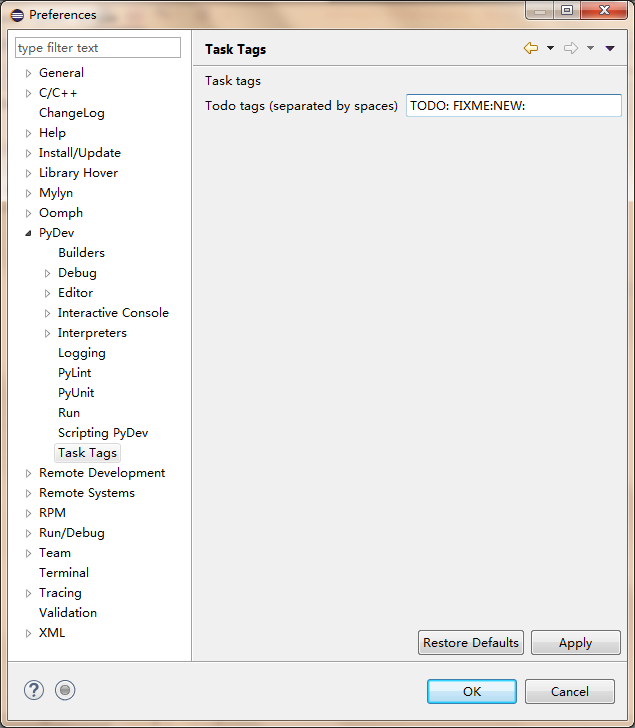
**配置方式**：进入Eclipse，通过Window → Preferences→PyDev→Task Tags进行配置，根据不同的任务类型，自定义标签。

图30 自定义任务标签

其中，TODO和FIXME是系统默认的两个标签符号，我们可以进行修改，或者增添新的标签，如笔者加入的NEW。此项可根据个人需要和习惯，对任务标签进行配置。

1. **添加任务标签注释**

我们可以在程序源码中加入使用任务标签的注释，以 “任务标签”关键词开头，PyDev用任务标签作为关键字，检索生成任务列表的注释。

1. **自动识别任务**

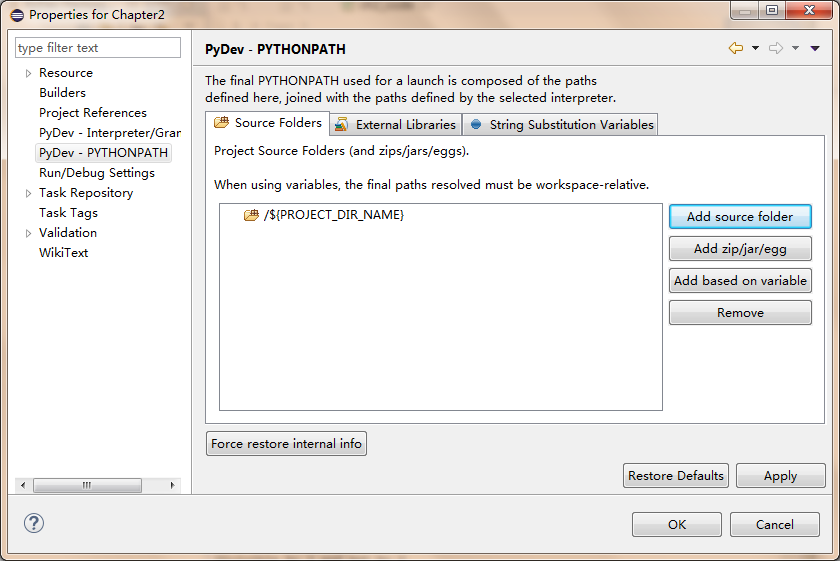
**配置方式：**首先需配置当前工程的PyDev-PYTHONPATH的source folder。Properties->PyDev-PYTHONPATH，点击“Add source folder”将源码目录加入。

图31 添加源码目录

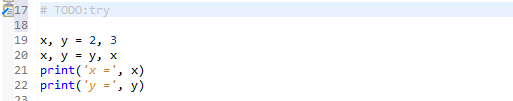
添加标签及注释，然后build当前项目（Ctrl+b）或者通过Project->Clean:

图32 自动识别标签

我们可以看见被标注行左侧出现了任务标识。

**特别说明：**任务标签的自动识别需要每个工程都进行单独配置一次。

1. **手动添加任务**

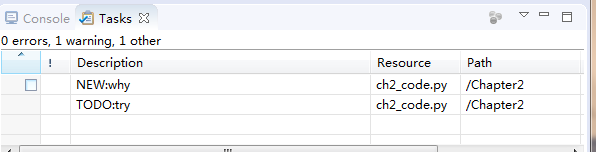
除了在上文中通过“#+”任务标签来添加以外，在PyDev环境下，系统还为用户提供了通过手动添加任务标识的方法——即在所需设置为任务行的最左侧，右键->Add Task，即可将其加入任务列表：

图33 任务列表

笔者手动添加任务标签NEW:why，顺利加进了任务列表中。

### 4.1.5语言规范静态检查

PyDev中集成了PEP8、和Pylint代码检查功能，然而这些功能默认状态都是关闭的。为了能够在程序开发过程中检查代码语法，我们需要在编写时需要注意规范静态检查。

**PEP8开启方法：**通过Window > Preferences>PyDev > Editor > Code Analysis > pep8.py，选择Errors/Warnings其中之一。

想要进行PEP8的检查，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。

为了方便我们使用，可以启动autopep8自动修改，因此还需要做一步：Windows -> Preferences ->输入 'autopep8' 作为搜索串。选择（Check）: Use autopep8.py for code formatting? 至此，PEP8的功能打开，在Python源码窗口，我们按 CTRL+SHIFT+F 就可以自动修改代码了。

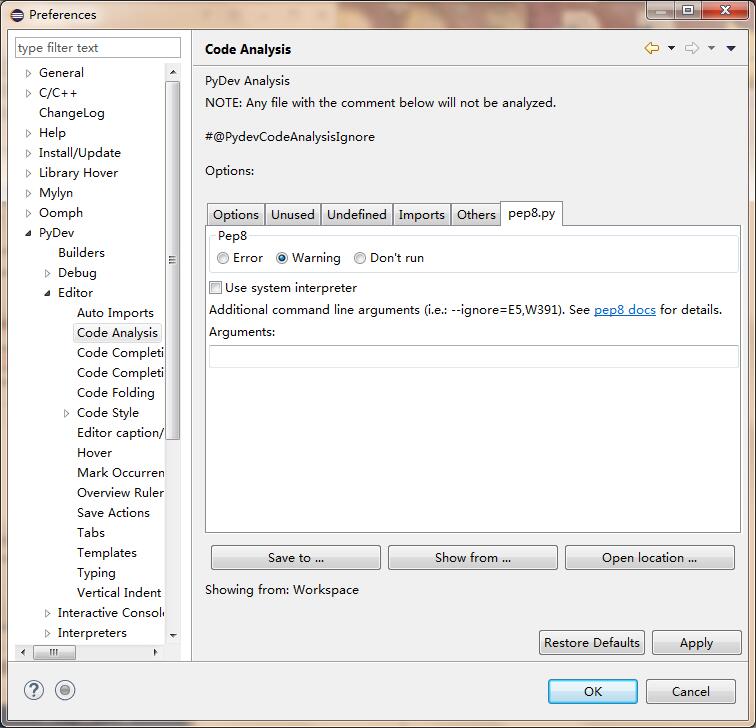
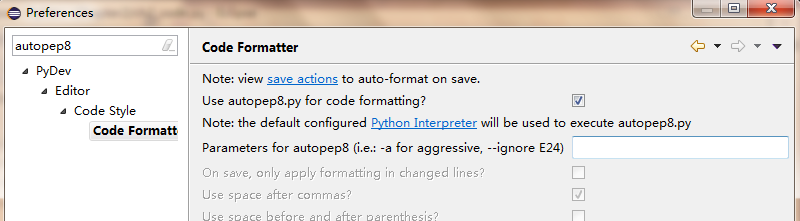
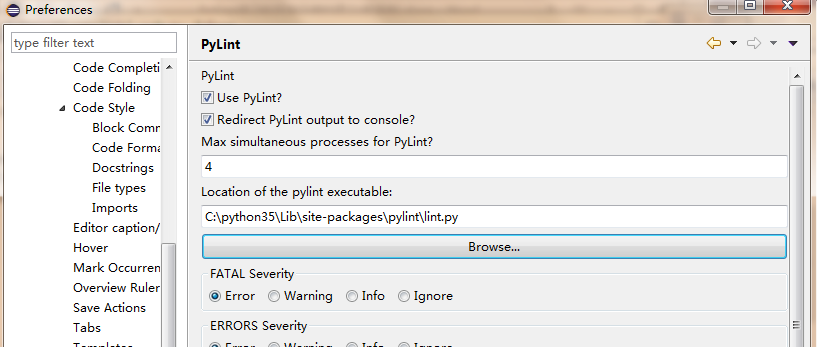
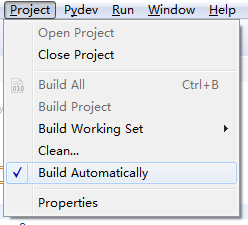
图34 开启pep8检查

图35 开启autopep8

**Pylint的开启方法：**通过Window -> preferences -> Pydev -> Pylint,选中"Use pylint?",“Browse…”找到安装好的lint.py的地址。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手动触发pylint。

图36 开启Pylint

图37 自动检查

## 4.2 Eclipse配置——文件字符编码方式

中文操作系统下，Eclipse默认工作空间编码方式为GBK, 这样的编码方式下，含中文字符的文件，在其他文本编辑器打开可能会乱码，因此，我们将配置编码为UTF-8。

### 4.2.1 工作空间编码方式

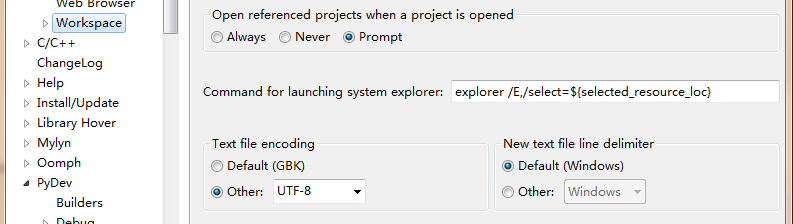
**配置方式：**Window->Preference，在弹出窗口中，点击general->workspace，修改Text file encoding为UTF-8。

图38 配置为UTF-8

### 4.2.2 工程编码方式

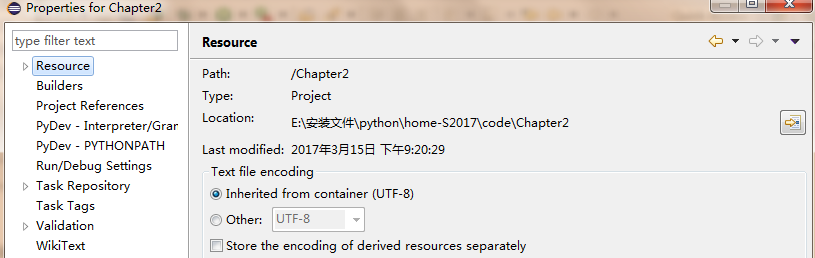
**配置方式：**点击右键工程，选择“properties”，选中“resources”， 修改“Text file encoding”为UTF-8(如果工作空间配置为UTF-8会继承过来)。

图39 工程编码配置

**特别说明：**如果配置到工作空间上，那个工作空间内的所有工程，都使用同样配置；如果工程需要自己的字符编码方式，那么，必须每个工程单独配置。

## 4.3 Eclipse使用示例

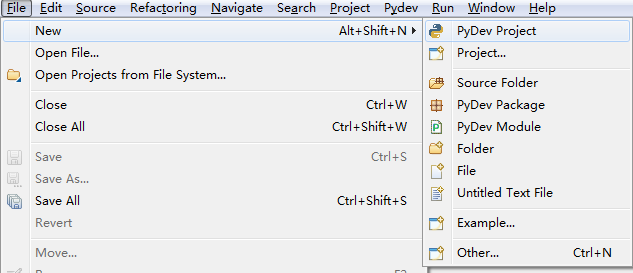
新建工程，存放在设定好的工作空间的目录下，方便查找和修改。

图40 新建工程

在新建的TEST工程下，依次新建PyDev Package和PyDev Module，然后我们就可以在新建的test.py的文件中编写代码了。

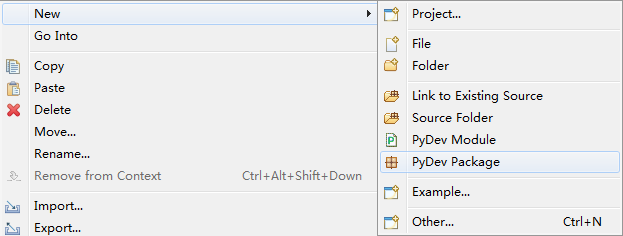
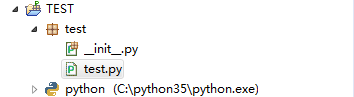
图41 创建 .py文件

图42 创建成功

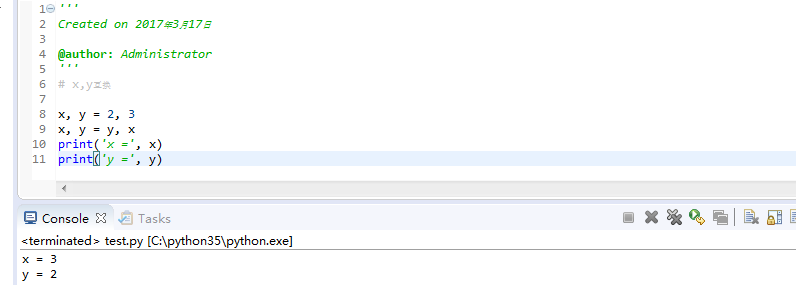
下图仅示意一个很简单的程序用以测试。

图43 简单程序

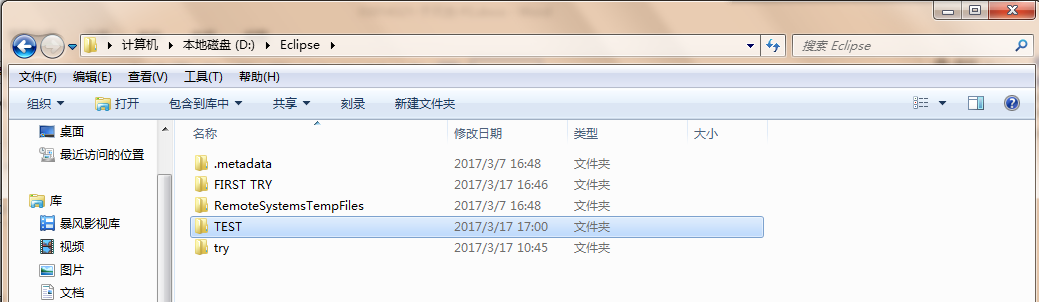
在对应的工作目录下，就会出现我们新建的工程。

图44 文件自动产生

**特别说明：**Eclipse的功能还有很多，我们可以通过查询Eclipse的内置help来学习和解答疑惑；另外，我们可以通过丰富Eclipse的插件，完善其功能。Eclipse默认安装对Markdown支持较弱，可以安装Markdown插件。Help->Eclipse Marketplace使用Markdown关键字可以找到2个插件：编辑和显示。包括Markdown Editor插件和GitHub Flavored Markdown Viewer插件。具体步骤参看Python学习和开发环境的建立(教学版).docx。

# 5 其他问题

在运行过程中遇到的问题在对应章节后的特别说明里大多已经提到了，并且提供了解决方案。除上述已经说明的问题外，其他需要特别注意的问题如下：

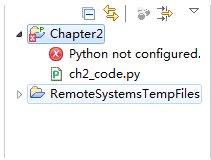
**问题1：**

图45 问题1

**原因：**没有配置基本的Python解释器。

**解决方法：**参见4.1.3章节配置解释器部分。

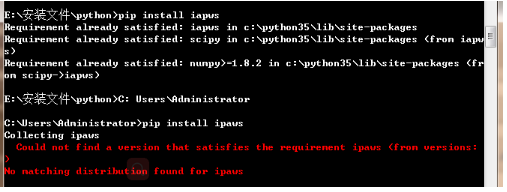
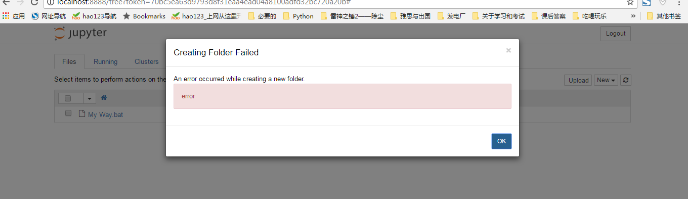
**问题2：**出现下图情况：

图46 问题2

**原因：**本以为是无法在C:盘目录中安装语言计算包，而实际上是拼写错误。

**解决方法：**pip install ipaws 改成 pip install iapws。

**问题3：**网页打开jupyter notebook后，无法新建文件。

图47 问题3

**原因：**如果基本配置都没有问题，可能和使用的浏览器软件有关（和网速没有关系，这是启动的是本地jupyter notebook web服务，无需联网）。

**解决方法：**重新启动jupyter ,使用微软的IE或者Edge。

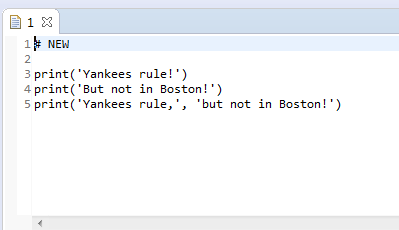
**问题4：**用Eclipse新建工程，在工程中新建文件编写代码，发现没有联想的功能，各代码之前没有区分度，任务标签无效果。

图48 问题4

**原因：**不是“.py”文件，编写代码无法使用语法检查和提示功能等。

**解决办法：**参见4.3章节Eclipse使用示例；或自己对新建的文件手动加“.py”。

# 6 个人小结

学习一门计算机语言和学习一门自然语言有相似之处，却也有很大的不同。计算机语言没有太多的词汇，但是注重逻辑和思维，在简单的思维构建上可以延伸出很多的功能，而自然语言的关键字、词很多，注重的是表达的涵义与流畅性。不过，语言都是一种工具，我们可以借一种语言学习、创造和表达，他们都是服务于我们本身的工具和方式。

Python号称“Battery Included"，其语言中包含了几乎所有的功能。初学Python，可能还没有感受到其功能的全面，但Python相比较其他不少主流编程语言，例如C语言、java，有更好的可读性，上手也相对容易。

在进行编写程序之前，我们进行了大量的准备工作——从语言开发环境的建立，拓展模块的安装，还包括交互计算环境Jupyter Notebook的建立和使用，以及通过最终建立的集成开发环境Eclipse的安装和配置，我们在准备过程中深深感受到其繁琐和复制，也体会到了Python这一语言背后的深厚的基础。

也正是这些复杂的流程让我对其开发环境有了一定的认识，学习一门语言最重要的可能不是学习如何编写代码，而是认识到其背后的核心点，我们在进行基本配置的过程虽然复杂，但是对于其机理有了初步的认识和了解，我想这也是老师让我们自己把这样的开发环境和开发工具建立起来的原因之一吧。

在学习过程中，我还认识到资源的重要性。Python是一门很开放的语言，很多人都在研究和学习，我们可以借助互联网、书籍等资源学到很多，在交流和互相补充之中我们得以进步和学习。

不过网上的教程和资料十分繁多，我们需要慎重选择，不可面面俱到；而且我认为，光看教程而不动手实践，是不可能真正掌握Python的精髓的。

# 7 参考资料

1. Python学习和开发环境的建立(教学版).docx

<https://github.com/PySEE/RecommendedPracticestree/S2016/P1>

1. Eclipse 4.3 (Kepler) for C/C++ Programming

<http://www3.ntu.edu.sg/home/ehchua/programming/howto/EclipseCpp_HowTo.html>

1. 为什么使用jupyter？，知乎

https://www.zhihu.com/question/37490497

1. Python 基础教程

http://www.runoob.com/python/python-tutorial.html

1. 廖雪峰的官方网站-Python教程

<http://www.liaoxuefeng.com/wiki/0014316089557264a6b348958f449949df42a6d3a2e542c000>