PCS1　交互环境之命令行

概述

Python 命令行，又称为 Python Shell，是默认的 Python 交互环境。

应用

进入 Python Shell

若在 Windows 下已经安装好 Python，也设置好了相应环境变量。在 GNU/Linux 下，通常情况是已经安装好 Python 了的，默认安装在/usr/bin/python 下，该路径已经放进你的 shell 搜索路径中。

打开 Windows 的命令行或 GNU/Linux 的终端，输入 python，即可进入 Python 交互式环境，界面如图 PCS 1-1 所示。

```
~$ python
Python 2.5.2 (r252:60911, Sep 27 2008, 18:55:47)
[GCC 4.1.3 20070929 (prerelease) (Ubuntu 4.1.2-16ubuntu2)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print 'hi'
hi
>>>
```

图 PCS 1-1

进入后，可以编写代码，调试，测试及查看相关帮助。若所做工作完成，退出命令行，可以使用 Ctrl+Z+Enter 键（Windows 下）或 Ctrl+D（GNU/Linux 下）键以 0 值退出（就是说，没有什么错误，正常退出）。如果这没有起作用，可以输入以下命令退出：import sys; sys.exit()。

使用交互环境

进入 Python Shell 后（如图 PCS 1-1 所示），我们来具体介绍如何使用该环境。

Python 解释器根据主提示符来执行，主提示符通常标识为三个“大于”号（>>>）；继续的部分被称为从属提示符，由三个点标识（...）。在第一行之前，解释器打印欢迎信息、版本号和授权提示如下：

```
Python 2.5.1 (r251:54863, Mar  7 2008, 04:10:12)
[GCC 4.1.3 20070929 (prerelease) (Ubuntu 4.1.2-16ubuntu2)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

输入多行结构时须用到从属提示符了。例如，下面这个 if 语句：

```
>>> sayhi = True
>>> if sayhi:
...     print 'hi~python!'
... else:
...     print 'say nothing!'
...
hi~python!
```

若在调试使用过程中有错误发生，则解释器会打印一个报错信息、栈跟踪器及出错位置等，便于修改。此为错误处理。

```
>>> if sayhi:
...     print 'hi~python!'
... else:
...
  File "<stdin>", line 4

    ^
IndentationError: expected an indented block
>>> import Mysqldb
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named Mysqldb
>>>
```

若在主提示符或附属提示符中输入中断符（Control-c，抛出一个 keyboardinterrupt 异常，它可以被 try 句截获）就会取消当前输入，回到主命令行。

在 Python Shell 中可以很方便地查看 Python 文档，包括类型、类库、模块等的使用资料。这些都是非常有用的。通过 help（"obj"）就可以看到 obj 的帮助信息，就像系统 Shell 中的 man 帮助一样，它提供了非常详细的资料。

小结

本文讲述了最基本的 Python 命令行使用方式，并描述了如何进入、使用、退出交互环境。这是非常简易的，很多情况下可以先在这里做些代码测试，通过后再进行脚本编辑。

练习

1. Python 是解释执行的，那么什么是解释执行呢？和其他语言（Java，C 等）的执行方式有什么区别？

相关参考

使用 Python 解释器：*http://doc.chinahtml.com/Manual/Python/tut/node4.html*