**Python 标识符**

在python里，标识符有字母、数字、下划线组成。

在python中，所有标识符可以包括英文、数字以及下划线（\_），但不能以数字开头。

python中的标识符是区分大小写的。

以下划线开头的标识符是有特殊意义的。以单下划线开头（\_foo）的代表不能直接访问的类属性，需通过类提供的接口进行访问，不能用"from xxx import \*"而导入；

以双下划线开头的（\_\_foo）代表类的私有成员；以双下划线开头和结尾的（\_\_foo\_\_）代表python里特殊方法专用的标识，如\_\_init\_\_（）代表类的构造函数。

**Python保留字符**

下面的列表显示了在Python中的保留字。这些保留字不能用作常数或变数，或任何其他标识符名称。

所有Python的关键字只包含小写字母。

|  |  |  |
| --- | --- | --- |
| and | exec | not |
| assert | finally | or |
| break | for | pass |
| class | from | print |
| continue | global | raise |
| def | if | return |
| del | import | try |
| elif | in | while |
| else | is | with |
| except | lambda | yield |

**行和缩进**

学习Python与其他语言最大的区别就是，Python的代码块不使用大括号（{}）来控制类，函数以及其他逻辑判断。python最具特色的就是用缩进来写模块。

缩进的空白数量是可变的，但是所有代码块语句必须包含相同的缩进空白数量，这个必须严格执行。如下所示：

if True:

print "True"

else:

print "False"

以下代码将会执行错误：

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

# 文件名：test.py

if True:

print "Answer"

print "True"

else:

print "Answer"

# 没有严格缩进，在执行时会报错

print "False"

执行以上代码，会出现如下错误提醒：

$ python test.py

File "test.py", line 5

if True:

^

IndentationError: unexpected indent

**IndentationError: unexpected indent** 错误是python编译器是在告诉你"Hi，老兄，你的文件里格式不对了，可能是tab和空格没对齐的问题"，所有python对格式要求非常严格。

如果是 **IndentationError: unindent does not match any outer indentation level**错误表明，你使用的缩进方式不一致，有的是 tab 键缩进，有的是空格缩进，改为一致即可。

因此，在Python的代码块中必须使用相同数目的行首缩进空格数。

建议你在每个缩进层次使用 **单个制表符** 或 **两个空格** 或 **四个空格** , 切记不能混用

**多行语句**

Python语句中一般以新行作为为语句的结束符。

但是我们可以使用斜杠（ \）将一行的语句分为多行显示，如下所示：

total = item\_one + \

item\_two + \

item\_three

语句中包含[], {} 或 () 括号就不需要使用多行连接符。如下实例：

days = ['Monday', 'Tuesday', 'Wednesday',

'Thursday', 'Friday']

**Python 引号**

Python 接收单引号(' )，双引号(" )，三引号(''' """) 来表示字符串，引号的开始与结束必须的相同类型的。

其中三引号可以由多行组成，编写多行文本的快捷语法，常用语文档字符串，在文件的特定地点，被当做注释。

word = 'word'

sentence = "这是一个句子。"

paragraph = """这是一个段落。

包含了多个语句"""

**Python注释**

python中单行注释采用 # 开头。

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

# 文件名：test.py

# 第一个注释

print "Hello, Python!"; # 第二个注释

输出结果：

Hello, Python!

注释可以在语句或表达式行末：

name = "Madisetti" # 这是一个注释

python 中多行注释使用三个单引号(''')或三个双引号(""")。

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

# 文件名：test.py

'''

这是多行注释，使用单引号。

这是多行注释，使用单引号。

这是多行注释，使用单引号。

'''

"""

这是多行注释，使用双引号。

这是多行注释，使用双引号。

这是多行注释，使用双引号。

"""

**Python空行**

函数之间或类的方法之间用空行分隔，表示一段新的代码的开始。类和函数入口之间也用一行空行分隔，以突出函数入口的开始。

空行与代码缩进不同，空行并不是Python语法的一部分。书写时不插入空行，Python解释器运行也不会出错。但是空行的作用在于分隔两段不同功能或含义的代码，便于日后代码的维护或重构。

记住：空行也是程序代码的一部分。

**等待用户输入**

下面的程序在按回车键后就会等待用户输入：

#!/usr/bin/python

raw\_input("\n\nPress the enter key to exit.")

以上代码中 ，"\n\n"在结果输出前会输出两个新的空行。一旦用户按下键时，程序将退出。

**同一行显示多条语句**

Python可以在同一行中使用多条语句，语句之间使用分号(;)分割，以下是一个简单的实例：

#!/usr/bin/python

import sys; x = 'runoob'; sys.stdout.write(x + '\n')

执行以上代码，输入结果为：

$ python test.py

runoob

**多个语句构成代码组**

缩进相同的一组语句构成一个代码块，我们称之代码组。

像if、while、def和class这样的复合语句，首行以关键字开始，以冒号( : )结束，该行之后的一行或多行代码构成代码组。

我们将首行及后面的代码组称为一个子句(clause)。

如下实例：

if expression :

suite

elif expression :

suite

else :

suite

**命令行参数**

很多程序可以执行一些操作来查看一些基本信，Python可以使用-h参数查看各参数帮助信息：

$ python -h

usage: python [option] ... [-c cmd | -m mod | file | -] [arg] ...

Options and arguments (and corresponding environment variables):

-c cmd : program passed in as string (terminates option list)

-d : debug output from parser (also PYTHONDEBUG=x)

-E : ignore environment variables (such as PYTHONPATH)

-h : print this help message and exit

[ etc. ]