

Python 基础教程

Python 基础教程

Python 简介

Python 环境搭建

Python 中文编码

Python 基础语法

Python 变量类型

Python 运算符

Python 条件语句

Python 循环语句

Python While 循环语句

Python for 循环语句

Python 循环嵌套

Python break 语句

Python continue 语句

Python pass 语句

Python Number(数字)

Python 字符串

Python 列表(List)

Python 元组

← Python 模块

Python File 方法 →

Python 文件I/O

本章只讲述所有基本的I/O函数，更多函数请参考Python标准文档。

打印到屏幕

最简单的输出方法是用print语句，你可以给它传递零个或多个用逗号隔开的表达式。此函数把你传递的表达式转换成一个字符串表达式，并将结果写到标准输出如下：

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

print "Python 是一个非常棒的语言，不是吗？";
```

你的标准屏幕上会产生以下结果：

```
Python 是一个非常棒的语言，不是吗？
```

读取键盘输入

Python提供了两个内置函数从标准输入读入一行文本，默认的标准输入是键盘。如下：

```
raw_input

input
```

raw\_input函数

raw\_input([prompt]) 函数从标准输入读取一个行，并返回一个字符串（去掉结尾的换行符）：

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

str = raw_input("请输入： ");
print "你输入的内容是：", str
```

这将提示你输入任意字符串，然后在屏幕上显示相同的字符串。当我输入"Hello Python !"，它的输出如下：

分类导航

HTML / CSS

JavaScript

服务端

数据库

移动端

XML 教程

ASP.NET

Web Service

开发工具

网站建设

Advertisement



反馈/建议

- Python 字典 (Dictionary)
- Python 日期和时间
- Python 函数
- Python 模块
- Python 文件 I/O
- Python File 方法
- Python 异常处理
- Python OS 文件/目录方法
- Python 内置函数
- Python 高级教程
  - Python 面向对象
  - Python正则表达式
  - Python CGI编程
  - Python MySQL
  - Python 网络编程
  - Python SMTP
  - Python 多线程
  - Python XML 解析
  - Python GUI 编程(Tkinter)
  - Python2.x与3.x版本区别
  - Python IDE
  - Python JSON
  - Python 100例

```
请输入: Hello Python!
你输入的内容是:  Hello Python!
```

input函数

input([prompt]) 函数和 raw\_input([prompt]) 函数基本类似，但是 input 可以接收一个Python表达式作为输入，并将运算结果返回。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

str = input("请输入: ");
print "你输入的内容是: ", str
```

这会产生如下的对应着输入的结果：

```
请输入: [x*5 for x in range(2,10,2)]
你输入的内容是:  [10, 20, 30, 40]
```

打开和关闭文件

现在，您已经可以向标准输入和输出进行读写。现在，来看看怎么读写实际的数据文件。  
Python 提供了必要的函数和方法进行默认情况下的文件基本操作。你可以用 **file** 对象做大部分的文件操作。

open 函数

你必须先用Python内置的open()函数打开一个文件，创建一个file对象，相关的方法才可以调用它进行读写。  
语法：

```
file object = open(file_name [, access_mode][, buffering])
```

各个参数的细节如下：

- file\_name：file\_name变量是一个包含了你要访问的文件名称的字符串值。
- access\_mode：access\_mode决定了打开文件的模式：只读，写入，追加等。所有可取值见如下的完全列表。这个参数是非强制的，默认文件访问模式为只读(r)。
- buffering:如果buffering的值被设为0，就不会有寄存。如果buffering的值取1，访问文件时会寄存行。如果将buffering的值设为大于1的整数，表明了这就是的寄存区的缓冲大小。如果取负值，寄存区的缓冲大小则为系统默认。

不同模式打开文件的完全列表：

模式	描述
r	以只读方式打开文件。文件的指针将会放在文件的开头。这是默认模式。
rb	以二进制格式打开一个文件用于只读。文件指针将会放在文件的开头。这

反馈/建议

	是默认模式。
r+	打开一个文件用于读写。文件指针将会放在文件的开头。
rb+	以二进制格式打开一个文件用于读写。文件指针将会放在文件的开头。
w	打开一个文件只用于写入。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。
wb	以二进制格式打开一个文件只用于写入。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。
w+	打开一个文件用于读写。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。
wb+	以二进制格式打开一个文件用于读写。如果该文件已存在则将其覆盖。如果该文件不存在，创建新文件。
a	打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。
ab	以二进制格式打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。
a+	打开一个文件用于读写。如果该文件已存在，文件指针将会放在文件的结尾。文件打开时会是追加模式。如果该文件不存在，创建新文件用于读写。
ab+	以二进制格式打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。如果该文件不存在，创建新文件用于读写。

File对象的属性

一个文件被打开后，你有一个file对象，你可以得到有关该文件的各种信息。  
以下是和file对象相关的所有属性的列表：

属性	描述
file.closed	返回true如果文件已被关闭，否则返回false。
file.mode	返回被打开文件的访问模式。
file.name	返回文件的名称。
file.softspace	如果用print输出后，必须跟一个空格符，则返回false。否则返回true。

如下实例：



```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# 打开一个文件
fo = open("foo.txt", "wb")
print "文件名: ", fo.name
print "是否已关闭 : ", fo.closed
print "访问模式 : ", fo.mode
print "末尾是否强制加空格 : ", fo.softspace
```

以上实例输出结果：

```
文件名:  foo.txt
是否已关闭 :  False
访问模式 :  wb
末尾是否强制加空格 :  0
```

## close()方法

File 对象的 close ( ) 方法刷新缓冲区里任何还没写入的信息，并关闭该文件，这之后便不能再进行写入。

当一个文件对象的引用被重新指定给另一个文件时，Python 会关闭之前的文件。用 close ( ) 方法关闭文件是一个很好的习惯。

语法：

```
fileObject.close();
```

例子：

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# 打开一个文件
fo = open("foo.txt", "wb")
print "文件名: ", fo.name

# 关闭打开的文件
fo.close()
```

以上实例输出结果：

```
文件名:  foo.txt
```

读写文件：

file对象提供了一系列方法，能让我们的文件访问更轻松。来看看如何使用read()和write()方法来读取和写入文件。

## write()方法

反馈/建议

write()方法可将任何字符串写入一个打开的文件。需要重点注意的是，Python字符串可以是二进制数据，而不是仅仅是文字。

write()方法不会在字符串的结尾添加换行符('\n')：

语法：

```
fileObject.write(string);
```

在这里，被传递的参数是要写入到已打开文件的内容。

例子：

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# 打开一个文件
fo = open("foo.txt", "wb")
fo.write( "www.runoob.com!\nVery good site!\n");

# 关闭打开的文件
fo.close()
```

上述方法会创建foo.txt文件，并将收到的内容写入该文件，并最终关闭文件。如果你打开这个文件，将看到以下内容：

```
$ cat foo.txt
www.runoob.com!
Very good site!
```

## read()方法

read( ) 方法从一个打开的文件中读取一个字符串。需要重点注意的是，Python字符串可以是二进制数据，而不是仅仅是文字。

语法：

```
fileObject.read([count]);
```

在这里，被传递的参数是要从已打开文件中读取的字节计数。该方法从文件的开头开始读入，如果没有传入count，它会尝试尽可能多地读取更多的内容，很可能是直到文件的末尾。

## 例子：

这里我们用到以上创建的 foo.txt 文件。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# 打开一个文件
fo = open("foo.txt", "r+")
str = fo.read(10);
print "读取的字符串是：" , str
```



反馈/建议

```
# 关闭打开的文件
fo.close()
```

以上实例输出结果：

```
读取的字符串是 : www.runoob
```

文件位置：

## 文件定位

tell()方法告诉你文件内的当前位置；换句话说，下一次的读写会发生在文件开头这么多字节之后。

seek ( offset [,from] ) 方法改变当前文件的位置。Offset变量表示要移动的字节数。From变量指定开始移动字节的参考位置。

如果from被设为0，这意味着将文件的开头作为移动字节的参考位置。如果设为1，则使用当前的位置作为参考位置。如果它被设为2，那么该文件的末尾将作为参考位置。

例子：

就用我们上面创建的文件foo.txt。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

# 打开一个文件
fo = open("foo.txt", "r+")
str = fo.read(10);
print "读取的字符串是 :", str

# 查找当前位置
position = fo.tell();
print "当前文件位置 :", position

# 把指针再次重新定位到文件开头
position = fo.seek(0, 0);
str = fo.read(10);
print "重新读取字符串 :", str

# 关闭打开的文件
fo.close()
```

以上实例输出结果：

```
读取的字符串是 : www.runoob
当前文件位置 : 10
重新读取字符串 : www.runoob
```

## 重命名和删除文件

Python的os模块提供了帮你执行文件处理操作的方法，比如重命名和删除文件。

反馈/建议

要使用这个模块，你必须先导入它，然后才可以调用相关的各种功能。

rename()方法：

rename()方法需要两个参数，当前的文件名和新文件名。

语法：

```
os.rename(current_file_name, new_file_name)
```

例子：

下例将重命名一个已经存在的文件test1.txt。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import os

# 重命名文件test1.txt到test2.txt。
os.rename( "test1.txt", "test2.txt" )
```

## remove()方法

你可以用remove()方法删除文件，需要提供要删除的文件名作为参数。

语法：

```
os.remove(file_name)
```

例子：

下例将删除一个已经存在的文件test2.txt。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import os

# 删除一个已经存在的文件test2.txt
os.remove("test2.txt")
```

## Python里的目录：

所有文件都包含在各个不同的目录下，不过Python也能轻松处理。os模块有许多方法能帮你创建，删除和更改目录。

### mkdir()方法

可以使用os模块的mkdir()方法在当前目录下创建新的目录们。你需要提供一个包含了要创建的目录名称的参数。

语法：

```
os.mkdir("newdir")
```



例子：

下例将在当前目录下创建一个新目录test。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import os

# 创建目录test
os.mkdir("test")
```

## chdir()方法

可以用chdir()方法来改变当前的目录。chdir()方法需要的一个参数是你想设成当前目录的目录名称。

语法：

```
os.chdir("newdir")
```

例子：

下例将进入"/home/newdir"目录。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import os

# 将当前目录改为"/home/newdir"
os.chdir("/home/newdir")
```

getcwd()方法：

getcwd()方法显示当前的工作目录。

语法：

```
os.getcwd()
```

例子：

下例给出当前目录：

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import os

# 给出当前的目录
print os.getcwd()
```

## rmdir()方法

反馈/建议



rmkdir()方法删除目录，目录名称以参数传递。

在删除这个目录之前，它的所有内容应该先被清除。

语法：

```
os.rmdir('dirname')
```

例子：

以下是删除" /tmp/test"目录的例子。目录的完全合规的名称必须被给出，否则会在当前目录下搜索该目录。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-

import os

# 删除"/tmp/test"目录
os.rmdir( "/tmp/test" )
```

## 文件、目录相关的方法

三个重要的方法来源能对Windows和Unix操作系统上的文件及目录进行一个广泛且实用的处理及操控，如下：

- File 对象方法: file对象提供了操作文件的一系列方法。
- OS 对象方法: 提供了处理文件及目录的一系列方法。

<div>在线实例</div> <ul style="list-style-type: none"><li>· <a href="#">HTML 实例</a></li><li>· <a href="#">CSS 实例</a></li><li>· <a href="#">JavaScript 实例</a></li><li>· <a href="#">Ajax 实例</a></li><li>· <a href="#">jQuery 实例</a></li><li>· <a href="#">XML 实例</a></li><li>· <a href="#">Java 实例</a></li></ul>	<div>字符集&amp;工具</div> <ul style="list-style-type: none"><li>· <a href="#">HTML 字符集设置</a></li><li>· <a href="#">HTML ASCII 字符集</a></li><li>· <a href="#">HTML ISO-8859-1</a></li><li>· <a href="#">HTML 实体符号</a></li><li>· <a href="#">HTML 拾色器</a></li><li>· <a href="#">JSON 格式化工具</a></li></ul>	<div>最新更新</div> <ul style="list-style-type: none"><li>· <a href="#">css 中的百分比...</a></li><li>· <a href="#">C++ 多线程</a></li><li>· <a href="#">C++ std::thread</a></li><li>· <a href="#">CSS3 rotation-p...</a></li><li>· <a href="#">Python中单线程...</a></li><li>· <a href="#">人生不如意事十...</a></li><li>· <a href="#">Bootstrap4 小工具</a></li></ul>	<div>站点信息</div> <ul style="list-style-type: none"><li>· <a href="#">意见反馈</a></li><li>· <a href="#">免责声明</a></li><li>· <a href="#">关于我们</a></li><li>· <a href="#">文章归档</a></li></ul>
---	---	---	---

关注微信



Copyright © 2013-2017 菜鸟教程  
runoob.com All Rights Reserved. 备案号：闽15012807号-1



反馈/建议

