Python3 字符串

字符串是 Python 中最常用的数据类型。 我们可以使用引号('或")来创建字符串。

创建字符串很简单，只要为变量分配一个值即可。例如：

```
var1 = 'Hello World!'
var2 = "Unigress"
```

Python 访问字符串中的值

Python 不支持单字符类型，单字符在 Python 中也是作为一个字符串使用。

Python 访问子字符串，可以使用方括号来截取字符串，如下实例：

```
#!/usr/bin/python3

var1 = 'Hello World!'
var2 = "Unigress"

print ("var1[0]: ", var1[0])
print ("var2[1:5]: ", var2[1:5])
```

以上实例执行结果：

```
var1[0]:  H
var2[1:5]:  nigr
```

Python字符串更新

你可以截取字符串的一部分并与其他字段拼接，如下实例：

```
#!/usr/bin/python3

var1 = 'Hello World!'

print ("已更新字符串 : ", var1[:6] + 'Unigress!')
```

以上实例执行结果

```
已更新字符串 :  Hello Unigress!
```

Python转义字符

在需要在字符中使用特殊字符时，python用反斜杠()转义字符。如下表：

转义字符	描述
(在行尾时)	续行符
\	反斜杠符号
'	单引号
"	双引号
\a	响铃
\b	退格(Backspace)
\e	转义
\000	空
\n	换行
\v	纵向制表符
\t	横向制表符
\r	回车
\f	换页
\oyy	八进制数，yy代表的字符，例如：\o12代表换行
\xyy	十六进制数，yy代表的字符，例如：\x0a代表换行
\other	其它的字符以普通格式输出

Python字符串运算符

下表实例变量a值为字符串 "Hello"，b变量值为 "Python"：

操作符	描述	实例
+	字符串连接	a + b 输出结果：HelloPython
*	重复输出字符串	a*2 输出结果：HelloHello
[]	通过索引获取字符串中字符	a[1] 输出结果 **e**
[:]	截取字符串中的一部分，遵循**左闭右开**原则，str[0,2] 是不包含第 3 个字符的。	a[1:4] 输出结果 **ell**
in	成员运算符 - 如果字符串中包含给定的字符返回 True	**'H' in a** 输出结果 True
not in	成员运算符 - 如果字符串中不包含给定的字符返回 True	**'M' not in a** 输出结果 True
r/R	原始字符串 - 原始字符串：所有的字符串都是直接按照字面的意思来使用，没有转义特殊或不能打印的字符。 原始字符串除在字符串的第一个引号前加上字母 r（可以大小写）以外，与普通字符串有着几乎完全相同的语法。	`print( r'\n' )` `print( R'\n' )`
%	格式字符串	请看下一节内容。

实例(Python 3.0+)

```
#!/usr/bin/python3

a = "Hello"
b = "Python"

print("a + b 输出结果: ", a + b)
print("a * 2 输出结果: ", a * 2)
print("a[1] 输出结果: ", a[1])
print("a[1:4] 输出结果: ", a[1:4])

if( "H" in a) :
    print("H 在变量 a 中")
else :
    print("H 不在变量 a 中")
```

```
if( "M" not in a) :
    print("M 不在变量 a 中")
else :
    print("M 在变量 a 中")

print (r'\n')
print (R'\n')
```

以上实例输出结果为：

```
a + b 输出结果:  HelloPython
a * 2 输出结果:  HelloHello
a[1] 输出结果:  e
a[1:4] 输出结果:  ell
H 在变量 a 中
M 不在变量 a 中
\n
\n
```

Python字符串格式化

Python 支持格式化字符串的输出 。尽管这样可能会用到非常复杂的表达式，但最基本的用法是将一个值插入到一个有字符串格式符 %s 的字符串中。

在 Python 中，字符串格式化使用与 C 中 sprintf 函数一样的语法。

实例

```
#!/usr/bin/python3

print ("我叫 %s 今年 %d 岁!" % ('小明', 10))
```

以上实例输出结果：

```
我叫 小明 今年 10 岁!
```

python字符串格式化符号：

符 号	描述
%c	格式化字符及其ASCII码
%s	格式化字符串
%d	格式化整数
%u	格式化无符号整型
%o	格式化无符号八进制数
%x	格式化无符号十六进制数
%X	格式化无符号十六进制数（大写）
%f	格式化浮点数字，可指定小数点后的精度
%e	用科学计数法格式化浮点数
%E	作用同%e，用科学计数法格式化浮点数
%g	%f和%e的简写
%G	%f 和 %E 的简写
%p	用十六进制数格式化变量的地址

格式化操作符辅助指令:

符号	功能
*	定义宽度或者小数点精度
-	用做左对齐
+	在正数前面显示加号(+)
	在正数前面显示空格
#	在八进制数前面显示零('0')，在十六进制前面显示'0x'或者'0X'(取决于用的是'x'还是'X')
0	显示的数字前面填充'0'而不是默认的空格
%	'%%'输出一个单一的'%'
(var)	映射变量(字典参数)
m.n.	m 是显示的最小总宽度,n 是小数点后的位数(如果可用的话)

Python2.6 开始，新增了一种格式化字符串的函数 str.format()，它增强了字符串格式化的功能。

Python三引号

python三引号允许一个字符串跨多行，字符串中可以包含换行符、制表符以及其他特殊字符。实例如下

实例

```
#!/usr/bin/python3

para_str = """这是一个多行字符串的实例
多行字符串可以使用制表符
TAB ( \t )。
也可以使用换行符 [ \n ]。
"""
print (para_str)
```

以上实例执行结果为：

```
这是一个多行字符串的实例
多行字符串可以使用制表符
TAB (	)。
也可以使用换行符 [
 ]。
```

三引号让程序员从引号和特殊字符串的泥潭里面解脱出来，自始至终保持一小块字符串的格式是所谓的WYSIWYG（所见即所得）格式的。

一个典型的用例是，当你需要一块HTML或者SQL时，这时用字符串组合，特殊字符串转义将会非常的繁琐。

```
errHTML = '''
<HTML><HEAD><TITLE>
Friends CGI Demo</TITLE></HEAD>
<BODY><H3>ERROR</H3>
<B>%s</B><P>
<FORM><INPUT TYPE=button VALUE=Back
ONCLICK="window.history.back()"></FORM>
</BODY></HTML>
'''
cursor.execute('''
CREATE TABLE users (
login VARCHAR(8),
uid INTEGER,
prid INTEGER)
''')
```

Unicode 字符串

在Python2中，普通字符串是以8位ASCII码进行存储的，**而Unicode字符串则存储为16位unicode字符串**，这样能够表示更多的字符集。使用的语法是在字符串前面加上前缀 **u**。

在Python3中，所有的字符串都是Unicode字符串。

Python 的字符串内建函数

Python 的字符串常用内建函数如下：

```
string = 'hello world'
string.capitalize() # 把字符串的第一个字符大写
string.center(width) # 返回一个原字符串居中,并使用空格填充至长度 width 的新字符串
string.count(str, beg=0, end=len(string)) # 返回 str 在 string 里面出现的次数，如果 beg 或者 end 指定则返回指定范围内 str 出现的次数
string.decode(encoding='UTF-8', errors='strict') # 以 encoding 指定的编码格式解码 string，如果出错默认报一个 ValueError 的 异 常 ， 除 非 errors 指 定 的 是 'ignore' 或 者'replace'
string.encode(encoding='UTF-8', errors='strict') # 以 encoding 指定的编码格式编码 string，如果出错默认报一个ValueError 的异常，除非 errors 指定的是'ignore'或者'replace'
string.endswith(obj, beg=0, end=len(string)) # 检查字符串是否以 obj 结束，如果beg 或者 end 指定则检查指定的范围内是否以 obj 结束，如果是，返回 True,否则返回 False.
string.expandtabs(tabsize=8) # 把字符串 string 中的 tab 符号转为空格，tab 符号默认的空格数是 8。
string.find(str, beg=0, end=len(string)) # 检测 str 是否包含在 string 中，如果 beg 和 end 指定范围，则检查是否包含在指定范围内，如果是返回开始的索引值，否则返回-1
string.index(str, beg=0, end=len(string)) # 跟find()方法一样，只不过如果str不在 string中会报一个异常.
string.isalnum() # 如果 string 至少有一个字符并且所有字符都是字母或数字则返回 True,否则返回 False
string.isalpha() # 如果 string 至少有一个字符并且所有字符都是字母则返回 True,否则返回 False
string.isdecimal() # 如果 string 只包含十进制数字则返回 True 否则返回 False.
string.isdigit() # 如果 string 只包含数字则返回 True 否则返回 False.
string.islower() # 如果 string 中包含至少一个区分大小写的字符，并且所有这些(区分大小写的)字符都是小写，则返回 True，否则返回 False
string.isnumeric() # 如果 string 中只包含数字字符，则返回 True，否则返回 False
string.isspace() # 如果 string 中只包含空格，则返回 True，否则返回 False.
string.istitle() # 如果 string 是标题化的(见 title())则返回 True，否则返回 False
string.isupper() # 如果 string 中包含至少一个区分大小写的字符，并且所有这些(区分大小写的)字符都是大写，则返回 True，否则返回 False
string.join(seq) # 以 string 作为分隔符，将 seq 中所有的元素(的字符串表示)合并为一个新的字符串
string.ljust(width) # 返回一个原字符串左对齐,并使用空格填充至长度 width 的新字符串
string.lower() # 转换 string 中所有大写字符为小写.
string.lstrip() # 截掉 string 左边的空格
string.maketrans(intab, outtab]) # maketrans() 方法用于创建字符映射的转换表，对于接受两个参数的最简单的调用方式，第一个参数是字符串，表示需要转换的字符，第二个参数也是字符串表示转换的目标。
max(str) # 返回字符串 str 中最大的字母。
min(str) # 返回字符串 str 中最小的字母。
string.partition(str) # 有点像 find()和 split()的结合体,从 str 出现的第一个位置起,把 字 符 串 string 分 成 一 个 3 元 素 的 元 组 (string_pre_str,str,string_post_str),如果 string 中不包含str 则 string_pre_str == string.
string.replace(str1, str2, num=string.count(str1)) # 把 string 中的 str1 替换成 str2,如果 num 指定，则替换不超过 num 次.
string.rfind(str, beg=0,end=len(string)) # 类似于 find()函数，不过是从右边开始查找.
string.rindex( str, beg=0,end=len(string)) # 类似于 index()，不过是从右边开始.
string.rjust(width) # 返回一个原字符串右对齐,并使用空格填充至长度 width 的新字符串
string.rpartition(str) # 类似于 partition()函数,不过是从右边开始查找.
string.rstrip() # 删除 string 字符串末尾的空格.
string.split(str="", num=string.count(str)) # 以 str 为分隔符切片 string，如果 num有指定值，则仅分隔 num 个子字符串
```

```
string.splitlines(num=string.count('\n')) # 按照行分隔，返回一个包含各行作为元素的列表，如果 num 指定则仅切片 num 个行.
string.startswith(obj, beg=0,end=len(string)) # 检查字符串是否是以 obj 开头，是则返回 True，否则返回 False。如果beg 和 end 指定值，则在指定范围内检查.
string.strip([obj]) # 在 string 上执行 lstrip()和 rstrip()
string.swapcase() # 翻转 string 中的大小写
string.title() # 返回"标题化"的 string,就是说所有单词都是以大写开始，其余字母均为小写(见 istitle())
string.translate(str, del="") # 根据 str 给出的表(包含 256 个字符)转换 string 的字符,要过滤掉的字符放到 del 参数中
string.upper() # 转换 string 中的小写字母为大写
string.zfill(width) # 返回长度为 width 的字符串，原字符串 string 右对齐，前面填充0
string.isdecimal() # isdecimal()方法检查字符串是否只包含十进制字符。这种方法只存在于unicode对象。
```

字符串截取字符补充：

```
#!/usr/bin/python3

# 0、a,b为参数。从字符串指针为a的地方开始截取字符，到b的前一个位置（因为不包含b）
var1 = "hello world"

#print(var1[a: b])
# 1、如果a,b均不填写，默认取全部字符。即，下面这两个打印结果是一样的
print(var1[: ])  # hello world
print(var1)      # hello world

# 2、如果a填写，b不填写（或填写的值大于指针下标），默认从a开始截取，至字符串最后一个位置
print(var1[3: ]) # lo world

# 3、如果a不填写，  b填写，默认从0位置开始截取，至b的前一个位置
print(var1[: 8]) # hello wo

# 4、如果a为负数，默认从尾部某一位置，开始向后截取
print(var1[-2: ]) # ld

# 5、如果a>=b，默认输出为空。
print(var1[3: 3])
```