Python3 字典dict

字典是另一种可变容器模型，且可存储任意类型对象。

字典的每个键值(key=>value)对用冒号(:)分割，每个对之间用逗号(,)分割，整个字典包括在花括号(**{}**)中 ,格式如下所示：

```
d = {key1 : value1, key2 : value2 }
```

键必须是唯一的，但值则不必。

值可以取任何数据类型，但键必须是不可变的，如字符串，数字或元组。

一个简单的字典实例：

```
dict = {'Alice': '2341', 'Beth': '9102', 'Cecil': '3258'}
```

也可如此创建字典：

```
dict1 = { 'abc': 456 };
dict2 = { 'abc': 123, 98.6: 37 };
```

访问字典里的值

把相应的键放入到方括号中，如下实例:

```
#!/usr/bin/python3

dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'}

print ("dict['Name']: ", dict['Name'])
print ("dict['Age']: ", dict['Age'])
```

以上实例输出结果：

```
dict['Name']:  Unigress
dict['Age']:  7
```

如果用字典里没有的键访问数据，会输出错误如下：

```
#!/usr/bin/python3

dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'};

print ("dict['Alice']: ", dict['Alice'])
```

以上实例输出结果：

```
Traceback (most recent call last):
  File "test.py", line 5, in <module>
    print ("dict['Alice']: ", dict['Alice'])
KeyError: 'Alice'
```

修改字典

向字典添加新内容的方法是增加新的键/值对，修改或删除已有键/值对如下实例:

```
#!/usr/bin/python3

dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'}

dict['Age'] = 8;               # 更新 Age
dict['School'] = "联航科技"     # 添加信息

print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
```

以上实例输出结果：

```
dict['Age']:  8
dict['School']:  联航科技
```

删除字典元素

能删单一的元素也能清空字典，清空只需一项操作。

显示删除一个字典用del命令，如下实例：

```
#!/usr/bin/python3

dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'}

del dict['Name'] # 删除键 'Name'
dict.clear()     # 清空字典
del dict         # 删除字典

print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
```

但这会引发一个异常，因为用执行 del 操作后字典不再存在：

```
Traceback (most recent call last):
  File "test.py", line 9, in <module>
    print ("dict['Age']: ", dict['Age'])
TypeError: 'type' object is not subscriptable
```

注：del() 方法后面也会讨论。

字典键的特性

字典值可以是任何的 python 对象，既可以是标准的对象，也可以是用户定义的，但键不行。

两个重要的点需要记住：

1）不允许同一个键出现两次。创建时如果同一个键被赋值两次，后一个值会被记住，如下实例：

```
#!/usr/bin/python3

dict = {'Name': 'Unigress', 'Age': 7, 'Name': '联航科技'}

print ("dict['Name']: ", dict['Name'])
```

以上实例输出结果：

```
dict['Name']:  联航科技
```

2）**键必须不可变**，所以可以用数字，字符串或元组充当，而用列表就不行，如下实例：

```
#!/usr/bin/python3

dict = {['Name']: 'Unigress', 'Age': 7}

print ("dict['Name']: ", dict['Name'])
```

以上实例输出结果：

```
Exception has occurred: TypeError
unhashable type: 'list'
  File "D:\python_study\1.py", line 3, in <module>
    dict = {['Name']: 'Unigress', 'Age': 7}
```

字典内置函数&方法

Python字典包含了以下内置函数：

序号	函数及描述	实例
1	len(dict) 计算字典元素个数，即键的总数。	`>>> dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'} >>> len(dict) 3`
2	str(dict) 输出字典，以可打印的字符串表示。	`>>> dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'} >>> str(dict) "{'Name': 'Unigress', 'Class': 'First', 'Age': 7}"`
3	type(variable) 返回输入的变量类型，如果变量是字典就返回字典类型。	`>>> dict = {'Name': 'Unigress', 'Age': 7, 'Class': 'First'} >>> type(dict) <class 'dict'>`

Python字典包含了以下内置方法：

序号	函数及描述
1	radiansdict.clear()删除字典内所有元素
2	radiansdict.copy()返回一个字典的浅复制
3	radiansdict.fromkeys() 创建一个新字典，以序列seq中元素做字典的键，val为字典所有键对应的初始值
4	radiansdict.get(key, default=None) 返回指定键的值，如果值不在字典中返回default值
5	key in dict如果键在字典dict里返回true，否则返回false
6	radiansdict.items()以列表返回可遍历的(键, 值) 元组数组
7	radiansdict.keys()返回一个迭代器，可以使用 list() 来转换为列表
8	radiansdict.setdefault(key, default=None)和get()类似, 但如果键不存在于字典中，将会添加键并将值设为default
9	radiansdict.update(dict2) 把字典dict2的键/值对更新到dict里
10	radiansdict.values()返回一个迭代器，可以使用 list() 来转换为列表
11	pop(key[,default]) 删除字典给定键 key 所对应的值，返回值为被删除的值。key值必须给出。 否则，返回default值。
12	popitem() 随机返回并删除字典中的一对键和值(一般删除末尾对)。

实例

clear(清空字典内容)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
print(stu.clear())

#输出：None
```

copy（拷贝字典）

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}

stu2 = stu.copy()
print(stu2)
```

```
print(stu2 is stu)
stu3 = stu
print(stu3)
print(stu3 is stu)
#输出{'num1': 'Tom', 'num2': 'Lucy', 'num3': 'Sam'}
#False
#{'num1': 'Tom', 'num2': 'Lucy', 'num3': 'Sam'}
#True
```

fromkeys(指定一个列表，把列表中的值作为字典的key,生成一个字典)

```
name = ['tom','lucy','sam']
print(dict.fromkeys(name))
print(dict.fromkeys(name,25))  #指定默认值

#输出: {'tom': None, 'lucy': None, 'sam': None}
#     {'tom': 25, 'lucy': 25, 'sam': 25}
```

get(指定key，获取对应的值)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
print(stu.get('num2'))

#输出: Lucy
```

items(返回由“键值对组成元素”的列表)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
print(stu.items())

#输出: dict_items([('num2', 'Lucy'), ('num3', 'Sam'), ('num1', 'Tom')])
```

keys(获取字典所有的key)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
print(stu.keys())

#输出: dict_keys(['num3', 'num1', 'num2'])
```

pop(获取指定key的value，并在字典中删除)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
name = stu.pop('num2')
print(name,stu)

#输出: Lucy {'num1': 'Tom', 'num3': 'Sam'}
```

popitem(随机获取某个键值对，并在字典中删除，一般删除末尾对)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
    'num4':'Jack'
}
name = stu.popitem()
print(name,stu)

#输出: ('num4', 'Jack') {'num1': 'Tom', 'num2': 'Lucy', 'num3': 'Sam'}
```

setdefault(获取指定key的value，如果key不存在，则创建)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
name = stu.setdefault('num5')
print(name,stu)

#输出: None {'num1': 'Tom', 'num2': 'Lucy', 'num3': 'Sam', 'num5': None}
```

update(添加键 - 值对到字典)

```
stu = {
    'num1':'Tom',
    'num2':'Lucy',
    'num3':'Sam',
}
stu.update({'num4':'Ben'})
print(stu)

#输出: {'num2': 'Lucy', 'num3': 'Sam', 'num1': 'Tom', 'num4': 'Ben'}
```