Python及其应用

主讲人：钱惠敏
E-mail: amandaqian@hhu.edu.cn

第3讲 字典

3.1 字典

3.2 字典的基本操作

3.3 字典的格式化

3.4 字典方法

3.1 字　典

- 字典是Python中唯一内建的映射类型：键→值。

- 映射（mapping）：通过名字（键）引用值的数据结构。键即为索引，键可以是数字、字符串甚至是元组。

- 字典是一种可变容器模型，且可存储任意类型对象。

3.1 字　典（续1）

➢字典的创建：

键值对

d = {key1 : value1, key2 : value2 }

注：

（1）空字典：{}。

（2）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。

```
>>> dict1 = {'小萌': '1001', '小智': '1002', '小强': '1003'}
>>> dict2 = { 'abc': 123, 98.6: 37 }
```

3.1 字　典（续2）

➢字典的创建（续）：

dict函数

```
>>> student=[('name','小萌'),('number','1001')]
>>> detail=dict(student)
>>> print('学生详细信息：',detail)
学生详细信息： {'name': '小萌', 'number': '1001'}
>>> print('学生姓名：',detail['name'])
学生姓名： 小萌
>>> print('学生学号：',detail['number'])
学生学号： 1001
```

3.2 字典的基本操作

- 字典的修改：增加新的键/值对，修改或删除已有键/值对。

```
>>> student={'小萌':'1001','小智':'1002','小强':'1003'}
>>> student['小强']='1005'  #更新小强的学号
>>> print('小强的学号是：%(小强)s' % student)
小强的学号是：1005
>>> student['小张']='1006' #添加一个学生
>>> print('小张的学号是：%(小张)s' % student)
小张的学号是：1006
```

3.2 字典的基本操作（续1）

➢删除字典元素或整个字典

```
>>> student={'小强': '1005', '小萌': '1001', '小智': '1002', '小张': '1006'}
>>> print('删除前:',student)
删除前: {'小强': '1005', '小萌': '1001', '小智': '1002', '小张': '1006'}
>>> del student['小张'] #删除键“小张”
>>> print('删除后:',student)
删除后: {'小强': '1005', '小萌': '1001', '小智': '1002‘}
```

3.2 字典的基本操作（续2）

➢ 字典的遍历

```
for key in dict:
    print (key + “:”+ str(stuendent[key]))
```

```
>>> student={'小强': '1005', '小萌': '1001', '小智': '1002', '小张': '1006'}
>>> for key in student:
>>>     print(key+”:”+str(student[key]))
小强:1005
小萌:1001
小智:1002
小张:1006
```

3.2 字典的基本操作（续3）

➢ 字典键的特性

（1）不允许同一个键出现两次。

```
>>> student={'小萌': '1001', '小智': '1002', '小萌': '1005'}
>>> print('学生信息：',student)
学生信息： {'小萌': '1005', '小智': '1002'}
```

（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。

```
>>> field={['name']:'小萌','number':'1001'}
Traceback (most recent call last):
  File "<pyshell#80>", line 1, in <module>
    field={['name']:'小萌','number':'1001'}
TypeError: unhashable type: 'list'
```

3.2 字典的基本操作（续3）

➢ len函数

len(dict)，该函数用于计算字典元素个数，即键的总数。

```
>>> student={'小萌': '1001', '小智': '1002', '小强': '1005','小张': '1006'}
>>> print('字典元素个数为：%d个' % len(student))
```

字典元素个数为：4个

➢ type函数

type(variable)，该函数返回输入的变量类型，如果输入变量是字典就返回字典类型。

```
>>> student={'小萌': '1001', '小智': '1002', '小强': '1005','小张': '1006'}
>>> print('字典的类型为：',type(student))
```

字典的类型为： <class 'dict'>

3.3 字典的格式化

字典的格式化方式

➢字典的格式化

```
>>> student={'小萌':'1001','小智':'1002','小强':'1003'}
>>> print('小强的学号是：%(小强)s' % student)
小强的学号是：1003
```

➢字典和列表的区别

dict的特点：

（1）查找和插入的速度极快，不会随着key的增加而变慢；

（2）需要占用大量的内存，内存浪费多。

List的特点是：

（1）查找和插入的时间随着元素的增加而增加；

（2）占用空间小，浪费内存很少。

3.4 字典方法

➢clear方法：删除字典内所有的项。

dict.clear()

```
>>> student={'小萌': '1001', '小智': '1002', '小强': '1005','小张': '1006'}
>>> print('字典元素个数为：%d个' % len(student))
字典元素个数为：4个
>>> student.clear()
>>> print('字典删除后元素个数为：%d个' % len(student))
字典删除后元素个数为：0个
```

3.4 字典方法(续1)

- copy方法：返回一个具有相同键/值对的新字典（这个方法是浅复制（shallow copy），因为值本身是相同的，而不是副本）。

  dict.copy()

```
>>> student={'小萌': '1001', '小智': '1002', '小强': '1005','小张': '1006'}
>>> st=student.copy()
>>> print('复制后得到的st为：',st)
复制后得到的st为： {'小强': '1005', '小萌': '1001', '小智': '1002', '小张': '1006'}
```

3.4 字典方法(续2)

- fromkeys方法：从序列创建一个新字典，以序列seq中元素做字典的键，value为字典所有键对应的初始值，缺省时初始值为None。

  dict.fromkeys(seq[, value])

```
>>> seq = ('name', 'age', 'sex')
>>> info = dict.fromkeys(seq)
>>> print ("新的字典为 : %s" %  info)
新的字典为 : {'name': None, 'sex': None, 'age': None}
>>> info = dict.fromkeys(seq, 10)
>>> print ("新的字典为 : %s" %  info)
新的字典为 : {'name': 10, 'sex': 10, 'age': 10}
```

3.4 字典方法(续3)

- get方法：返回指定键key的值，如果值不在字典中，返回默认值default。

  dict.get(key, default=None)

```
>>> student={'小萌': '1001', '小智': '1002'}
>>> print ('小萌的学号为: %s' %  student.get('小萌'))
小萌的学号为: 1001
```

3.4 字典方法(续4)

- in操作符：用于判断键key是否存在于字典dict中，如果键在字典里，返回true，否则返回false。

  key in dict

```
>>> student={'小萌': '1001', '小智': '1002'}
>>> print('小萌在student字典中：%s'%('小萌' in student))
小萌在student字典中：True
>>> print('小强在student字典中：%s'%('小强' in student))
小强在student字典中：False
```

3.4 字典方法(续5)

- items方法：以列表返回可遍历的(键, 值) 元组数组。

  dict.items()

```
>>> student={'小萌': '1001', '小智': '1002'}
>>> print('调用items方法的结果：%s'% student.items())
调用items方法的结果：dict_items([('小萌', '1001'), ('小智', '1002')])
```

3.4 字典方法(续6)

➢keys方法：以列表返回一个字典dict的所有的键。

dict.keys()

```
>>> student={'小萌': '1001', '小智': '1002'}
>>> print('字典student所有的键为：%s'% student.keys())
字典student所有的键为：dict_keys(['小萌', '小智'])
```

3.4 字典方法(续7)

➢values方法：以列表形式返回字典中的所有值。

dict.values()

```
>>> student={'小萌': '1001', '小智': '1002','小李':'1001'}
>>> print('student字典所有值为：%s'% list(student.values()))
student字典所有值为：['1001', '1001', '1002']
```

3.4 字典方法(续8)

- setdefault方法：获得与给定键key相关联的值，如果键不存在于字典中，将会添加键并将值设为默认值default 。

dict.setdefault(key, default=None)

```
>>> student={'小萌': '1001', '小智': '1002'}
>>> print('小强的键值为：%s'% student.setdefault('小强'))
小强的键值为：None
>>> print('小智的键值为：%s'% student.setdefault('小智'))
小智的键值为：1002
>>> print('student字典新值为：%s'% student)
student字典新值为：{'小强': None, '小萌': '1001', '小智': '1002'}
```

3.4 字典方法(续9)

➢update方法把字典dict2的键/值对更新到dict里。

dict.update(dict2)

```
>>> student={'小萌': '1001', '小智': '1002'}
>>> student2={'小李':'1003'}
>>> print('原student字典为：%s'% student)
原student字典为： {'小萌': '1001', '小智': '1002'}
>>> student.update(student2)
>>> print('新student字典为：%s'% student)
新student字典为： {'小萌': '1001', '小智': '1002', '小李': '1003'}
```