

Python语言基础与应用

基本扩展模块 / 持久化模块: `shelve`

陈斌 北京大学 gischen@pku.edu.cn



持久化模块

- › 对象持久化
- › 构造数据库
- › 常用操作

对象持久化

临时性对象

类创建的对象并不是真正的数据库记录

存储在内存而不是文件中

关闭python，实例将消失



对象持久化

对象在创建它们的程序退出之后依然存在

对象持久化

› 标准库模块

- **pickle**

任意Python对象格式化和解格式化

- **dbm**

实现一个可通过键访问的文件系统，以存储字节串

- **shelve**

按照键把pickle处理后的对象存储到一个文件中

构造数据库

› shelve模块

提供基本的存储操作，通过构造一个简单的数据库，像操作字典一样按照键存储和获取本地的Python对象，使其可以跨程序运行而保持持久化

› 键

必须是字符串，且是唯一的

› 值

任何类型的Python对象



构造数据库

› 与字典类型的区别

一开始必须打开shelve，并且在修改后需要关闭它

› 数据处理

不支持类似SQL的查询工具

但只要通过键获取到保存在文件的对象，就可以像正常的数据对象一样处理

常用操作

› 将任何数据对象，保存到文件中去

```
d = shelve.open(filename)
```

open函数在调用时返回一个shelf对象，通过该对象可以存储内容

› 类似字典形式访问，可读可写

```
d[key] = data
```

```
value = d[key]
```

```
del d[key]
```

› 操作完成后，记得关闭文件

```
d.close()
```

常用操作

无作用

```
import shelve

d = shelve.open(filename)  # open -- filename
                             # library

d[key] = data               # store data
                             # using an editor
data = d[key]              # retrieve a value
                             # if no such key
del d[key]                 # delete data
                             # if no such key

flag = key in d             # true if the key is in the database
klist = list(d.keys())     # a list of keys

# as d was opened WITHOUT writeback=True
d['xx'] = [0, 1, 2]        # this works
d['xx'].append(3)          # *this does not work

# having opened d without writeback=True
temp = d['xx']              # extracts the value
temp.append(5)              # mutates the list
d['xx'] = temp              # stores the new value

# or, d=shelve.open(filename,writeback=True)
# d['xx'].append(5) and have it work as expected
# consume more memory and make the d.close()

d.close()                  # close it
```