Python基础技术问题总结

2015/03/25 · [基础知识](http://python.jobbole.com/category/basic/) · [2 评论](http://python.jobbole.com/81233/#article-comment)· [Python](http://python.jobbole.com/tag/python/)

分享到：**9**

原文出处： [liuaiqi627 的博客](http://www.pulpcode.cn/study-note/2015/02/26/python-technical-summary/)

**Introduction**

此篇文章总结python一些常见的技术问题,每一个都是简短的解释,篇幅不大，不断更新中…

**解释一下python的 and-or 语法**

python的and-or常常被用来实现类C语言中的三元运算符:?, 更骚包的写法是 xxx and xxx or xxx and xxx or xxx，连接多个表达式，一行就够了。

Python



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | if a > 10:      return "big"  elif a > 5:      return "middle"  else:      return "small"    a > 5 and "big" or a > 3 and "middle" or "small" |

但是要注意and-or并不完全等价于三元运算符，因为 a and b or c ,如果b是个假值，那么将返回c。

不过有技巧能够实现：

Python



|  |  |
| --- | --- |
| 1  2 | def choose(bool, a, b):      return (bool and [a] or [b])[0] |

**Python里面如何拷贝一个对象？**

这个问题在 python cookbook中有很好的解释。

标准库中的copy模块提供了两个方法来实现拷贝.一个方法是copy,它返回和参数包含内容一样的对象.

Python



|  |  |
| --- | --- |
| 1  2 | import copy  new\_list = copy.copy(existing\_list) |

有些时候,你希望对象中的属性也被复制,可以使用deepcopy方法:

Python



|  |  |
| --- | --- |
| 1  2 | import copy  new\_list\_of\_dicts = copy.deepcopy(existing\_list\_of\_dicts) |

Python



|  |  |
| --- | --- |
| 1  2  3  4  5 | copy(x)  Shallow copy operation on arbitrary Python objects.    deepcopy(x, memo=None, \_nil=[])  Deep copy operation on arbitrary Python objects. |

**python中如何判断对象相等**

也有一个类似的问题会被面试官问爆，我们不能将这个问题弄混了。

首先是C#中字符串的==和equal方法。

**“==” :**

对于内置值类型而言， == 判断两个内存值是否相等。

对于用户自定义的值类型而言(Struct)， == 需要重载，否则不能使用。

对于引用类型而言，默认是同一引用才返回true，但是系统重载了很多引用类型的 == （比如下文提到的string），所以c#中引用类型的比较并不建议使用 ==。

**“equals” :**

对于值类型而言， 内存相等才返回true。

对于引用类型而言，指向同一个引用才算相等。

但是比较特殊的是字符串String,是一个特殊的引用型类型，在C#语言中，重载了string的equals()方法，使string对象用起来就像是值类型一样。

**python中的 ==**

python中的对象包含三要素:id, type, value

id 用来标识唯一一个对象，type标识对象的类型，value用来设置对象的值。

is 判断是否是一个对象，使用id来判断的。

== 是判断a对象的值是否是b对象的值，默认调用它的\_\_eq\_\_方法。

**什么是lambda函数？它有什么好处?**

早期自认为看了许多东西，总想把lambda表达式和lambda演算之间的关系研究明白，但是后来发现，文献中除了说lambda这个名字的由来是借鉴了lambda演算之外，再没有谈及其它。

那么lambda表达式有什么好处？匿名函数，一般用来给filter，map这样的函数式编程服务，map(lambda x: x\*2,[1,2,3,4,5])

或者就是一次性产品。类似于这种即用即扔的产品：(lambda x: x\*2)(3)

或者作为回调函数，传递给某些应用，比如消息处理。

**请写出一段Python代码实现删除一个list里面的重复元素**

可能有人用循环对比删除的办法去实现，不过有一个更优雅的办法，list(set(li))。

**命名技巧**

今天阅读代码，发现一个不错的函数命名方式:

Python



|  |  |
| --- | --- |
| 1 | def request(\_argv): |

就是把所有的参数前面都加上\_下划线，这样你在函数体中，一眼就可以看出那些是局部变量，那些是作为参数传入的，类似把全局变量前面加上g。