**Python学习笔记：Python字符编码问题的经验小结**

2017年02月23日 23:00:56

阅读数：783

Python的字符编码问题是一个很大的坑，特别是对于Python 2.X而言。造成这个问题的主要原因是我们没有统一的编码。

**0x01 编码操作**

这里我不讲编码的原理，简单易懂的可以看这里：[字符串和编码](http://www.liaoxuefeng.com/wiki/001374738125095c955c1e6d8bb493182103fac9270762a000/001386819196283586a37629844456ca7e5a7faa9b94ee8000)   
Python中的编码，其实可以粗分为两大类：Unicode和非Unicode。相互之间可以利用encode和decode进行转换。

#coding:utf-8

#前面限定了是utf-8

utf8\_str='test'

#unicode字符串

unicode\_str=u'测试'

#相互转换

utf8\_str.decode('utf-8')

unicode\_str.encode('utf-8')

**0x02 编码识别**

对于一段字符串，我们有时候并不好确定它真实的编码方式，最简单的办法就是使用chardet进行识别。

安装：

pip install chardet

使用起来也很简单：

#coding:utf-8

from chardet import detect

import urllib2

html=urllib2.urlopen('http://www.baidu.com').read()

print detect(html)

结果：

{'confidence': 0.99, 'encoding': 'utf-8'}

* 1

confidence显示的是别正确的概率。encoding是识别的结果。

**0x03 需要注意的问题**

（1）对字符串进行操作，比如拼接等，需要编码格式一致。这个很好理解，就不再赘述；   
（2）当需要保存文本文件的时候，有时候并不知道是什么编码方式，建议使用二进制方式：

import urllib2

html=urllib2.urlopen('http://www.baidu.com').read()

with open('baidu.txt','wb') as f:

f.write(html)

（3）有些特殊的字符串可能非常奇怪，导致Python转换过程中无法处理，这种情况我们可以使用decode的特殊处理方法：

#coding:utf-8

test='u123\x01\u0000特殊'

print test.decode('utf-8','ignore')

（4）Windows系统的编码平台可能比较特殊，导致容易出现乱码，我们可以对字符串的输入输出这么处理：

import sys

#输出打印:info是unicode编码

def new\_print(info):

print info.encode(sys.stdin.encoding)

#解码为unicode编码

def new\_input(info):

print info.decode(sys.stdin.encoding)

**0x04 后记**

Python字符编码确实是一个让人头疼的问题，但是只要使用正确的方式，还是可以较好解决的。希望上面的内容对大家有所帮助。