5.14 忽略文件名编码¶

问题¶

你想使用原始文件名执行文件的I/O操作，也就是说文件名并没有经过系统默认编码去解码或编码过。

解决方案¶

默认情况下，所有的文件名都会根据 `sys.getfilesystemencoding()` 返回的文本编码来编码或解码。比如：

```
>>> sys.getfilesystemencoding()
'utf-8'
>>>
```

如果因为某种原因你想忽略这种编码，可以使用一个原始字节字符串来指定一个文件名即可。比如：

```
>>> # Wrte a file using a unicode filename
>>> with open('jalape\xf1o.txt', 'w') as f:
...     f.write('Spicy!')
...
6
>>> # Directory listing (decoded)
>>> import os
>>> os.listdir('.')
['jalapeño.txt']

>>> # Directory listing (raw)
>>> os.listdir(b'.') # Note: byte string
[b'jalapen\xcc\x83o.txt']

>>> # Open file with raw filename
>>> with open(b'jalapen\xcc\x83o.txt') as f:
...     print(f.read())
...
Spicy!
>>>
```

正如你所见，在最后两个操作中，当你给文件相关函数如 `open()` 和 `os.listdir()` 传递字节字符串时，文件名的处理方式会稍有不同。

讨论¶

通常来讲，你不需要担心文件名的编码和解码，普通的文件名操作应该就没问题了。 但是，有些操作系统允许用户通过偶然或恶意方式去创建名字不符合默认编码的文件。 这些文件名可能会神秘地中断那些需要处理大量文件的Python程序。

读取目录并通过原始未解码方式处理文件名可以有效的避免这样的问题， 尽管这样会带来一定的编程难度。

关于打印不可解码的文件名，请参考5.15小节。