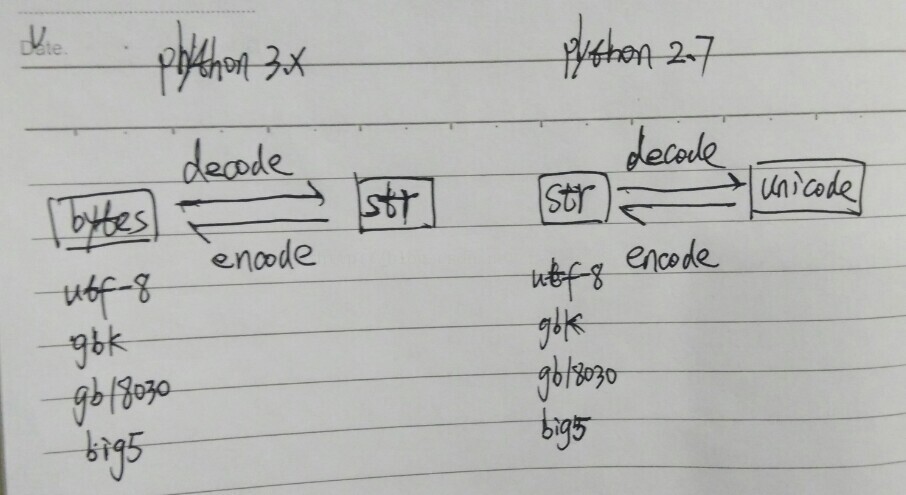
**python 2. x和3.x中的字符串区别**



如图2.x和3.x的区别：

2.x中字符串有str和unicode两种类型，str有各种编码区别，unicode是没有编码的标准形式。unicode通过编码转化成str，str通过解码转化成unicode。

**3.x中将字符串和字节序列做了区别:**

字符串str是字符串标准形式与2.x中unicode类似，bytes类似2.x中的str有各种编码区别。bytes通过解码转化成str，str通过编码转化成bytes…

Bytes字节码只有decode方法变为str, str只有encode方法变为bytes.

**Python3 的decode方法：**

decode () 方法以指定的编码格式解码 bytes 对象。默认编码为 'utf-8'。

语法

decode ()方法语法：

bytes.decode(encoding="utf-8", errors="strict")

参数

encoding -- 要使用的编码，如"UTF-8"。

errors -- 设置不同错误的处理方案。默认为 'strict’, 意为编码错误引起一个UnicodeError。 其他可能得值有 'ignore', 'replace', 'xmlcharrefreplace', 'backslashreplace' 以及通过 codecs.register\_error() 注册的任何值。

返回值

该方法返回解码后的字符串。

实例

以下实例展示了decode()方法的实例：

实例(Python 3.0+)

#!/usr/bin/python3

str = "菜鸟教程";

str\_utf8 = str.encode(encoding="UTF-8")

str\_gbk = str.encode(encoding="GBK")

print(str)

print ("UTF-8 编码：", str\_utf8)

print ("GBK 编码：", str\_gbk)

print ("UTF-8 解码：", str\_utf8.decode('UTF-8','strict'))

print ("GBK 解码：", str\_gbk.decode('GBK','strict'))

以上实例输出结果如下：

菜鸟教程

UTF-8 编码： b'\xe8\x8f\x9c\xe9\xb8\x9f\xe6\x95\x99\xe7\xa8\x8b'

GBK 编码： b'\xb2\xcb\xc4\xf1\xbd\xcc\xb3\xcc'

UTF-8 解码： 菜鸟教程

GBK 解码： 菜鸟教程

**Python3 的bytes对中英文的不同显示方式：**

b’ABC’ 🡪 建立一个以ASCII码编码的ABC字节码

b’ABC’[0] 🡪 输出为65，不是’A’, 是英文A的ASCII码编码。

b’你好’ 🡪 无法建立，报错。因为默认的ASCII码无法将你好顺利编码成ASCII码的字节码。

'ABC'.encode(encoding="utf-8") 🡪 显示为b’ABC’, 虽然本质是字节码但是全是英文，Python会以b告诉是字节码，但是显示出内容

'你好’. encode(encoding="utf-8") 🡪显示为b’ \xe4\xbd\xa0\xe5\xa5\xbd'，由于含有非英文，所以Python只会显示出编码。

**Python3 使用bytes或者str写文件：**

当使用python写文件的时候，如果以”b”的方式写文件，也就是用二进制写文件，则需要写入bytes,也就是f.write(b”xxx”), 否则报错。

当使用python写文件的时候，如果不以”b”的方式写文件，也就是不用二进制写文件，则需要写入str,也就是f.write(“xxx”) ， 否则报错。

**Python3使用telnet模块：**

Python3使用telnet模块的时候:

Write方法需要写入bytes类型。

Read方法默认读出的为bytes类型

**Python3使用subprocess模块：**

在我们使用subprocess模块的时候，除非指定了universal\_newlines为true或者在subprocess方法中指定了encoding参数，否则返回的是bytes类型。