字符的编码表

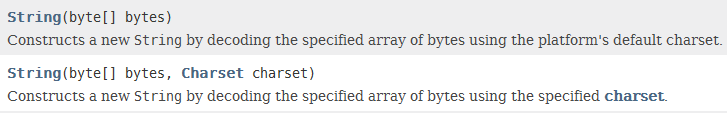
1. **编码**：由**字符**到**字节**，需要编码字符集，即编码表；

**解码**：由**字节**到**字符，需要解码字符集，即解码表**。

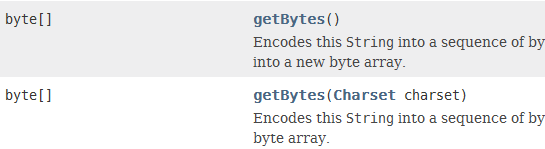
解码表与编码表都是一回事，都是字符集。解码和编码的字符表一致时，才可以正确获取数据。

1. 利用String中的编码和解码问题：

**（1）构造方法**：（编码过程）利用默认的编码表或指定字符集。



（2）**解码过程**：



示例：

String str = **new** String("联通");

System.***out***.println("解码"+str);

**byte**[] bytes = str.getBytes();

**byte**[] bytes2 = str.getBytes("utf-8");

System.***out***.println(Arrays.*toString*(bytes));

System.***out***.println(Arrays.*toString*(bytes2));

String str1 = **new** String(bytes,"gbk");

String str2 = **new** String(bytes2,"utf-8");

System.***out***.println(str1);

System.***out***.println(str2);

1. **转换流：属于处理流。**

**字符流 = 字节流+ 编码表。 输入字节流InputStream读取字节，然后利用译码表解码，得到相应的字符（本质也是二进制文件）；再根据编码表编码，得到字节。**

1. **OutputStreamWriter**：输出流，**输入字节流**转换成**输入字符流**，编码。
2. **InputStreamReader**：输入流，**输出字符流**转换成**输出字节流**，解码。
3. 验证： **InputStreamReader**是先把字节流读取的字节+译码表，生成字符，然后OutputStreamWriter利用字符流读取字符+编码表，生成字节，字节流输出。

示例：

**public** **static** **void** main(String[] args) **throws** IOException {

File file = **new** File("e:\\aaa.txt");

File file2 = **new** File("e:\\bbb.txt");

BufferedReader buffread = **new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(file),"gbk"));

BufferedWriter buffwriter = **new** BufferedWriter(**new** OutputStreamWriter(**new** FileOutputStream(file2),"utf-8" ));

String line = **null**;

**while**(**null** != (line = buffread.readLine())){

System.***out***.println(line);

buffwriter.write(line);

buffwriter.newLine();

}

buffread.close();

buffwriter.close();

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

BufferedReader buffread2 = **new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(file2),"gbk"));

**while**(**null** != (line = buffread2.readLine())){

System.***out***.println(line);

}

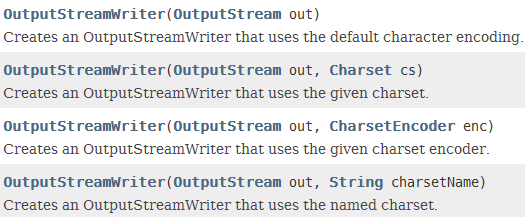
buffread2.close();

}

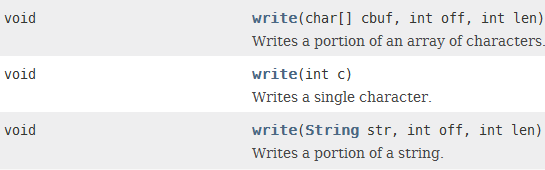
结果：



1. **OutputStreamWriter：继承于java.io.Writer。**
   1. **构造方法：与Writer的区别就是过了个字符集。**



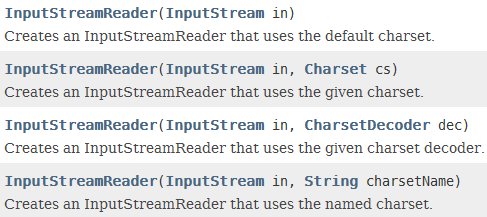
* 1. 方法：
     1. **write方法**：



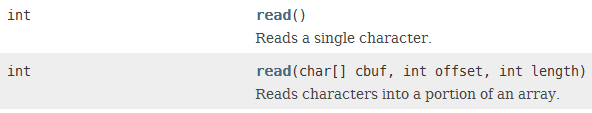
* + 1. getEncoding方法



1. **InputStreamReader：继承于java.io.Reader。**
   1. 构造方法：



* 1. 成员方法：
     1. read方法：



* + 1. getEncoding方法:\

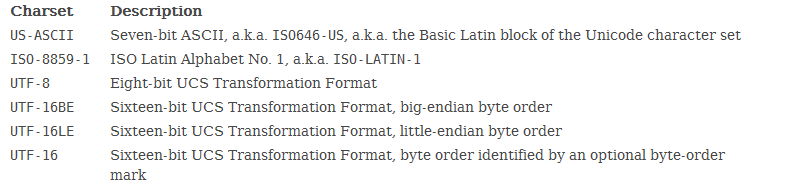


1. **常见的编码表**：
2. **ASCII ：利用一个字节的7位表示字符。**

**American Standard Code for Information Interchange**，[**美国标准**](http://baike.baidu.com/view/555380.htm)**信息交换代码。**

ASCII 码使用指定的**7 位**或**8 位**[二进制数](http://baike.baidu.com/view/1349.htm)组合来表示128 或256 种可能的[字符](http://baike.baidu.com/view/263416.htm) **。分为标准ASCII和扩展ASCII。标准ASCII 是现今通用的单字节编码系统。0-127 是7位ASCII 码的范围，是国际标准。而扩展ASCII不是国际标准。**

在**标准ASCII**中，其最高位(b7)用作[**奇偶校验位**](http://baike.baidu.com/view/492663.htm)。所谓奇偶校验，是指在代码传送过程中用来检验是否出现错误的一种方法，一般分[奇校验](http://baike.baidu.com/view/93325.htm)和偶校验两种。[奇校验](http://baike.baidu.com/view/93325.htm)规定：正确的代码一个[字节](http://baike.baidu.com/view/60408.htm)中1的个数必须是奇数，若非奇数，则在最高位b7添1；偶校验规定：正确的代码一个字节中1的个数必须是[偶数](http://baike.baidu.com/view/20858.htm)，若非偶数，则在最高位b7添1。

1. **Unicode：所有字符都使用2个字节表示一个字符。国际标准码，融入许多国家的文字。Java语言使用的就是Unicode码。**
2. **UTF-8：使用1个或2个或3个字节表示一个字符。**
3. **GB2312：中国的中文编码表。**
4. **GBK：使用2个字节表示一个中文汉字。升级的中国中文编码表，融入了更多的中文文字字符。**
5. **ISO8859-1：拉丁码表，欧洲码表。用1个字节表示。（8位）**
6. **还有UTF-16、UTF-16BE、UTF-16LE等。**
7. 
8. 需要用到编码表的地方（主要是两个：**转换IO流**和**字符与字节转换**）：
9. 转换流: InputStreamReader 和 OutputStreamWriter。
   1. InputStreamReader：

使用默认的编码表。









* 1. **OutputStreamWriter：**

使用默认的**编码表**。





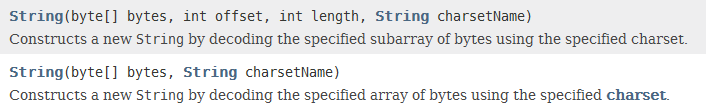




1. 字节和字符之间的转换：
   1. **字节-🡪字符**：解码。利用**String的构造方法**，在参数列表中传入指定的码表。

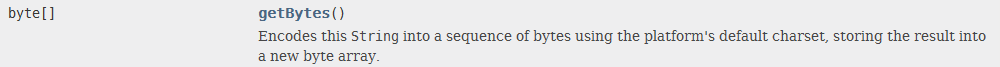


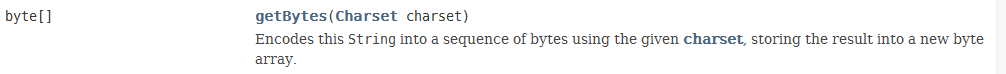


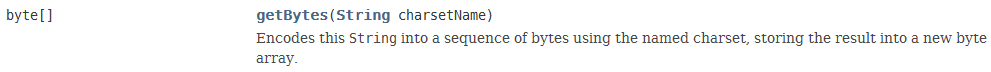


* 1. **字符-🡪字节：**

**没有指定编码表，则是利用平台默认的编码表。**







1. 在IO流中，如果涉及到需要**指定的编码表**，必须使用**转换流**才可以即InputStreamReader和OutputStreamWriter。如：

InputStreamReader isr = new InputStreamReader(new FileInputStream("e:\\copy.txt"),"GBK");

OutputStreamWriter osw=new OutputStreamWriter(new FileOutputStream("e:\\copy.txt"),"UTF-8");即只有InputStreamReader和OutputStreamWriter中**可以指定编码表**，因为只有字节和字符之间转换才需要**编码表**。

例子：

BufferedWriter bufw = new

BufferedWriter(new OutputStreamWriter(new FileOutputStream("e:\\utf-8.txt"),"UTF-8"));

BufferedReader bufr = new

BufferedReader(new InputStreamReader(new FileInputStream("e:\\utf-8.txt"),"UTF-8"));

bufw.write("山东科技大学");

bufw.flush();

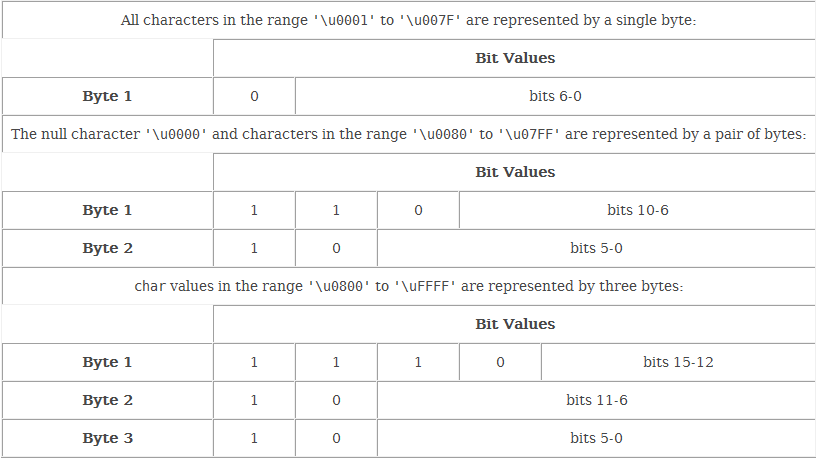
String line = bufr.readLine();

System.out.println(line);// 山东科技大学

bufw.close();

bufr.close();

1. UTF-8：



在使用UTF-8时，根据字节的前缀判断是1个字节的编码还是2个或者3个。如果字节的最高位为0，则是一个字节的编码；如果是110开头且紧接着第二个是10开头，则是2个字节的编码；如果第一个字节是以1110开头且第二个和第三个是以10开头，则是3个字节的编码。如果这3种情况都不满足，则会修改字节值，使其不可复原。

1. 问题分析：

一般客户端都有自己平台的编码表，但是服务端也是有自己的编码表，如果客户端向服务端发送GBK编码的**字节数据**，服务端利用UTF-8进行**解码**，然后再利用UTF-8编码，把**字节数据**返回客户端，客户端再利用**GBK解码**，发现结果错误。出现错误的原因是UTF-8进行解码和重新编码过程中，会修改原码，导致错误。

例子：

package IOTest;

import java.io.IOException;

import java.io.UnsupportedEncodingException;

public class Coding {

public static void main(String[] args) throws IOException {

String str = new String("abc中华人民共和国");

byte[] bes = str.getBytes("GBK");

byte[] besgbk = bianAndJieCode(bes,"GBK");

byte[] besuni = bianAndJieCode(bes,"Unicode");

byte[] besutf = bianAndJieCode(bes,"UTF-8");

String strgbk = new String(besgbk,"GBK");

System.out.println("str4 ： "+ strgbk);

String struni = new String(besuni,"GBK");

System.out.println("str5 ："+ struni);

String strutf = new String(besutf,"GBK");

System.out.println("str6 ： "+strutf);

}

private static byte[] bianAndJieCode(byte[] bes,String charset1)

throws UnsupportedEncodingException {

String str3 = new String(bes,charset1);//解码

System.out.println(str3);

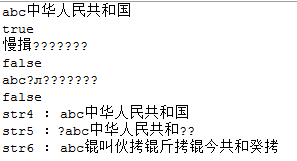
byte[] bes3 = str3.getBytes(charset1);//再次编码

System.out.println(bes.length == bes3.length);

return bes3;

}

}



1. 练习题：
2. GBK编码：每个汉字都是两个字节，而每个英文字母都是1个字节，若放在一起，如何实现正确获取字符呢？

同样是一句**字符串**"ab好孩子gk"，解码时GBK解出10个字节，而UTF-8解出13个字节，因为字母都是一个字节，而汉字在GBK中占2个字节，而在UTF-8 中占3个字节。

String str = new String("ab好孩子gk");

byte[] bs1 = str.getBytes("GBK");

byte[] bs2 = str.getBytes("UTF-8");

System.out.println(bs1.length);//10

System.out.println(bs2.length);//13

代码：（默认GBK中一个汉字对应的两个字节都是负数）

package IOTest;

import java.io.IOException;

public class getChac {

public static void main(String[] args) throws IOException {

String str = new String("ab好孩子gk");

byte[] bs = str.getBytes("GBK");

for(int num = 1;num <= bs.length;num++) {

String strr = readNumByte(bs, num );

System.out.println("截取"+num+"个字节的结果 ：\t"+strr);

}

}

private static String readNumByte(byte[] bs,int num) throws IOException {

int count=0;

for (int i = num-1; i >= 0; i--) {

//从后面往前数，出现正字节就退出

if(bs[i] < 0)

count++;

else

break;

}

// for (int i = 0; i < num; i++) {

// //从前往后数，把num个字节都判断一遍，显然没有上面的方法好

// if(bs[i]<0)

// count++;

// }

String strr =null;

if(count%2 != 0)

strr = new String(bs,0,num-1,"GBK");

else

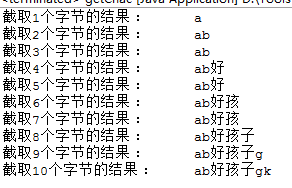
strr = new String(bs,0,num,"GBK");

return strr;

}

}

结果为:



1. 若为**UTF-8编码**呢？（默认一个汉字对应的三个字节都是负数）

代码：

package IOTest;

import java.io.IOException;

public class getChac2 {

public static void main(String[] args) throws IOException {

String str = new String("ab好孩子gk");

byte[] bs = str.getBytes("UTF-8");

for(int num = 1;num <= bs.length;num++) {

String strr = readNumByte(bs, num );

System.out.println("截取"+num+"个字节的结果 ：\t"+strr);

}

}

private static String readNumByte(byte[] bs,int num) throws IOException {

int count=0;

for (int i = num-1; i >= 0; i--) {

//从后面往前数，出现正字节就退出

if(bs[i] < 0)

count++;

else

break;

}

// for (int i = 0; i < num; i++) {

// //从前往后数，把num个字节都判断一遍，显然没有上面的方法好

// if(bs[i]<0)

// count++;

// }

String strr =null;

count %= 3;

if(count == 0)

strr = new String(bs,0,num,"UTF-8");

else if(count == 1)

strr = new String(bs,0,num-1,"UTF-8");

else

strr = new String(bs,0,num-2,"UTF-8");

return strr;

}

}

结果为：

