

非贷款,0元入学,不1万就业不给1分钱学费,我们已干四年了!

笔记总链接: http://bbs.itheima.com/thread-200600-1-1.html

8, IO 8.2 IO流

需求:文件切割器。

代码: 01.

06. 07.

08. 09.

8.2.11 IO包中的其他类

import java.io.File; import java.io.FileInputStream; 02. 03. import java.io.FileOutputStream; import java.io.IOException; 04. import java.util.Properties; 05.

public class SplitFileDemo{

private static final int SIZE = 1024*1024; 10. 11. public static void main(String[] args) throws IOException{ File file = new File("0.mp3"); 12. 13. splitFile(file); } 15. public static void splitFile(File file) throws IOException { 16. //用读取流关联源文件 17. FileInputStream fis = new FileInputStream(file); 18. 19. //定义一个1M的缓冲区 byte[] buf = new byte[SIZE];

20. 22. //创建目的 23. FileOutputStream fos = null; 24. 25. 26. int len = 0; 27. int count = 1; 28. //切割文件时,必须记录住被切割文件的名称,以及切割出来碎片文件的个数,以方 29. 便于合并。 //这个信息为了进行描述,使用键值对的方式,用到了properties对象。 30. 31. Properties prop = new Properties(); 32. 33. File dir = new File("c:\\partFiles"); 34. 35. if(!dir.exists()) 36. dir.mkdirs(); 37. while((len = fis.read(buf)) != -1){ 38. fos = new FileOutputStream(new File(dir,(count++) + 39. ".part")); 40. fos.write(buf,0,len); 41. fos.close(); } 42. 43. //将被切割文件的信息保存到prop集合中 44. prop.setProperty("partcount",count + ""); 45. prop.setProperty("filename",file.getName()); 47. 48. fos = new FileOutputStream(new File(dir,count + ".properties")); 49.

//将prop集合中的数据存储到文件中 50. prop.store(fos, "save file info"); 51. 52. fis.close(); fos.close(); } 55. 56. } 复制代码 运行结果: - - X ■ 管理员: C:\Windows\system32\cmd.exe D:\code\day24>javac SplitFileDemo.java D:\code\day24>java SplitFileDemo → → 计算机 → 本地磁盘 (D:) → code → day24 44 文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H) 共享▼ 组织▼ 包含到库中▼ 新建文件夹 ■ 图片 修改日期 名称 类型

立 文档 0.mp3 2015/6/7 15:29 MP3 文件 × 🍏 0.mp3 属性 常规 安全 详细信息 以前的版本 0.mp3 文件类型: MP3 文件 (.mp3) 打开方式: 🎏 酷我音乐盒 更改(C)... D:\code\day24 位置: 大小: 4.87 MB(5,117,408 字节) 4.88 MB (5,120,000 字节) 占用空间:

(C) □ □ → 計算机 → 本地磁盘 (C:) → partFiles

共享▼

___ 1.part

2.part

3.part

4.part

5.part

6.properties

名称

新建文件夹

文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

Ε

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

#Sun Jun 07 21:40:03 CST 2015

import java.io.File;

import java.io.FileInputStream;

包含到库中 ▼

■ 图片

文档

音乐

■ 计算机

→ 迅雷下载

☑ 6.properties - 记事本

#save file info

filename=0.mp3 partcount=6

需求:文件合并器。

代码:

01.

02.

03.

22. 23.

24.

25.

26.

27.

28. 29.

30. 31.

32.

33. 34.

35.

44.

45.

46.

47.

48. 49. 50.

51.

52.

53. 54. 搜索

▼ ধ 搜索 partF

类型

PART 文件

PART 文件

PART 文件

PART 文件

PART 文件

PROPERTIES 文件

- - X

修改日期

2015/6/7 21:40

2015/6/7 21:40

2015/6/7 21:40

2015/6/7 21:40

2015/6/7 21:40

2015/6/7 21:40

import java.io.FileOutputStream; import java.io.FilenameFilter; 04. import java.io.IOException; 05. import java.io.SequenceInputStream; 06. 07. import java.util.ArrayList; 08. import java.util.Collections; 09. import java.util.Enumeration; import java.util.Iterator; 10. import java.util.Properties; 11. 12. 13. public class MergeFile{ public static void main(String[] args) throws IOException { File dir = new File("c:\\partFiles"); 15. 16. mergeFile(dir); 17. } 18. public static void mergeFile(File dir) throws IOException { 19. 20. //获取指定目录下的配置文件对象 21.

File[] files = dir.listFiles(new SuffixFilter(".properties"));

throw new RuntimeException(dir + ",该目录下没有properties扩

throw new RuntimeException("碎片文件不符合要求,个数不对!应该

ArrayList<FileInputStream> al = new ArrayList<FileInputStream>();

al.add(new FileInputStream(partFiles[x-1]));

final Iterator<FileInputStream> it = al.iterator();

36. String filename = prop.getProperty("filename"); 37. 38. int count = Integer.parseInt(prop.getProperty("partcount")); 39. //获取该目录下的所有碎片文件 40. File[] partFiles = dir.listFiles(new SuffixFilter(".part")); 43. if(partFiles.length != (count - 1)){

//将碎片文件和流对象关联并存储到集合中

for(int x = 1; x <= partFiles.length; x++){</pre>

Properties prop = new Properties();

FileInputStream fis = new FileInputStream(confile);

if(files.length!=1)

//记录配置文件对象

//获取该文件中的信息

prop.load(fis);

File confile = files[0];

展名的文件或者不唯一");

是" + count + "个");

}

}

55. //将多个流合并成一个序列流 56. 57. Enumeration<FileInputStream> en = Collections.enumeration(al); 58. 59. SequenceInputStream sis = new SequenceInputStream(en); 60. FileOutputStream fos = new FileOutputStream(new File(dir,filename)); 62. byte[] buf = new byte[1024*1024]; 63. 64. int len = 0; 65. 66. 67. while((len = sis.read(buf)) != -1){ fos.write(buf,0,len); 68. 69. } 70. 71. fos.close(); 72. sis.close(); } 73. 74. } 75. 76. class SuffixFilter implements FilenameFilter{ private String suffix; 77. 78. 79. public SuffixFilter(String suffix){ super(); 80. this.suffix = suffix; 81. 82. } 83. 84. public boolean accept(File dir,String name){ return name.endsWith(suffix); 85. 86. } 87. 复制代码 运行结果: _ D X ■ 管理员: C:\Windows\system32\cmd.exe D:\code\day24>javac MergeFile.java D:\code\day24>java MergeFile
 →
 →
 计算机 →
 本地磁盘 (C:) →
 partFiles
 ¥ 44 文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H) 新建文件夹 组织▼ 打开 图片 名称 修改日期 类型 文档 **№** 0.mp3 2015/6/7 21:46 MP3 文件 X **》 0.mp3 属性** 详细信息 以前的版本 常规 安全 0.mp3 MP3 文件 (.mp3) 文件类型: 更改(C)... 打开方式: 🍲 酷我音乐盒 C:\partFiles 位置: 4.87 MB(5,117,408 字节) 大小: 操作对象 ObjectInputStream与ObjectOutputStream P.S. 被操作的对象需要实现Serializable。

类通过实现java.io.Serializable接口以启用序列化功能,Serializable只是一个标记接口。

示例1:

import java.io.Serializable;

class Person implements Serializable{

public Person(String name,int age){

public void setName(String name){

this.name = name;

public void setAge(int age){

this.age = age;

this.name = name;

this.age = age;

public String getName(){

return name;

public int getAge(){ return age;

private String name;

private int age;

01.

02.

03.

04.

05. 06. 07.

08.

09.

10.

11.

12.

13. 14. 15.

16.

17. 18.

19. 20.

21. 22.

24.

25. 26.

27.

运行结果:

组织▼

示例2:

17.

18.

运行结果:

小强:22

Person.java - 记事本

}

■ 管理员: C:\Windows\system32\cmd.exe

D:\code\day24>javac ObjectStreamDemo.java

如果修改Person类中属性的修饰符为public,就会报如下异常。

D:\code\day24>java ObjectStreamDemo

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

class Person implements Serializable{

this.name = name; this.age = age;

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day24>javac Person.java

3)

732)

原因分析:

API解释如下:

常。

01.

02. 03. 04.

06.

07.

08.

09. 10.

11.

12.

14.

15. 16.

17. 18. 19.

20. 21.

22. 23.

24.

25. 26.

27.

}

运行结果:

nu11:0

特点:

P.S.

示例1:

01.

02. 03.

04.

RandomAccessFile

1. 该对象即能读,又能写。

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day24>javac ObjectStreamDemo.java

通过skipBytes(int x),seek(int x)等方法来达到随机访问。

4. 其实该对象就是将字节输入流和输出流进行了封装。

RandomAccessFile不是io体系中的子类。

import java.io.IOException;

import java.io.RandomAccessFile;

D:∖code∖day24>javac RandomAccessFileDemo.java

共享▼

ranacc.txt

名称

新建文件夹

D:\code\day24>java RandomAccessFileDemo

文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

包含到库中 ▼

组织▼

🌟 收藏夹 ▶ 下载

🧻 ranacc.txt - 记事本

运行结果:

5. 该对象的源或者目的只能是文件。通过构造函数就可以看出。

2. 该对象内部维护了一个byte数组,并通过指针可以操作数组中的元素。

3. 可以通过getFilePointer方法获取指针的位置,和通过seek方法设置指针的位置。

D:\code\day24>java ObjectStreamDemo

随机访问文件,自身具备读写的方法。

复制代码

}

}

}

}

D:\code\day24>java ObjectStreamDemo

D:\code\day24>javac ObjectStreamDemo.java

serialVersionUID = -7079755575125947015

认的计算值,但是数组类没有匹配 serialVersionUID 值的要求。

class Person implements Serializable{

public String name ; public int age ;

public Person(String name, int age){

this.name = name;

this.age = age;

public String getName(){

return name ;

public int getAge(){

return age ;

public void setAge(int age){

writeObject方法不能写入类及其所有超类型的瞬态和静态字段的值。

this.age = age;

public void setName(String name){ this.name = name;

Exception in thread "main" java.io.InvalidClassException: Person; local class in compatible: stream classdesc serialVersionUID = -3273783136645942309, local clas

at java.io.ObjectStreamClass.initNonProxy(ObjectStreamClass.java:562) at java.io.ObjectInputStream.readNonProxyDesc<ObjectInputStream.java:158

at java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1496) at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:1

at java.io.ObjectInputStream.readObjectO<ObjectInputStream.java:1329> at java.io.ObjectInputStream.readObject(ObjectInputStream.java:351)

如果可序列化类未显式声明 serialVersionUID,则序列化运行时将基于该类的各个方面计算该类的默认 serialVersionUID 值,如 "Java(TN) 对象序列化规范"中所还。不过,*强烈建议* 所有可序列化类都显式声明 serialVersionUID 值,原因是计算默认的 serialVersionUID 对类的详细信息具有较高的敏感性,根据编译器实现的不同可能千差万别,这样在反序列化过程中可能会导致意外的 InvalidClassException。因此,为保证 serialVersionUID 值跨不同 java 编译器实现的一致性,序列化类必须声明一个明确的 serialVersionUID 值。还强烈建议使用 private 修饰符显示声明 serialVersionUID(如果可能),原因是这种声明仅应用于直接声明类—— serialVersionUID 字段作为继承成员没有用处。数据类不能声明一个明确的 serialVersionUID,因此它们总是具有默 11 的计算法。但是数据表现是是一个表面图象。可以是是一个表面图象。

为Person类添加序列号属性。此时,再将Person类中属性修饰符修改为public,也不会出现任何异

private static final long serialVersionUID = 9527;

at ObjectStreamDemo.readObj(ObjectStreamDemo.java:12) at ObjectStreamDemo.main(ObjectStreamDemo.java:6)

Serializable:用于给被序列化的类加入ID号,用于判断类和对象是否是同一个版本。

public Person(String name,int age){

import java.io.Serializable;

public String name; private int age;

复制代码

☆ 收藏夹

🏮 下载

}

}

}

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day24>javac Person.java

D:\code\day24>java ObjectStreamDemo

文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

打开▼

D:\code\day24>javac ObjectStreamDemo.java

🕌 ▶ 计算机 ▶ 本地磁盘 (D:) ▶ code ▶ day24

新建文件夹

obj.object

名称

复制代码

H

搜索

_ 0 X

v 44

类型

- 0 X

0

- - X

2015/6/7 21:49 OBJECT 文件

修改日期

搜索。

import java.io.FileInputStream; 01. import java.io.ObjectInputStream; 02. public class ObjectStreamDemo{ 04. public static void main(String[] args) throws Exception { 05. readObj(); 06. } 07. 08. 09. public static void readObj() throws Exception { ObjectInputStream ois = new ObjectInputStream(new 10. FileInputStream("obj.object")); 11. 12. Person p = (Person)ois.readObject(); 13. System.out.println(p.getName() + ":" + p.getAge()); 14. 15. 16. ois.close();

示例3: import java.io.Serializable; 01. 02. 03. class Person implements Serializable{ 04. private static final long serialVersionUID = 9527; 06. private transient String name; 07. private static int age; 08. 09. public Person(String name, int age){ 10. this.name = name; this.age = age; 11. 12. } 13. public String getName(){ 14. return name; 15. 16. } 17. public void setName(String name){ 20. } 21. 22. public int getAge(){ 23. return age; 24. } 25. public void setAge(int age){ 27. this.age = age; 28. } 29. } 复制代码

- - X

44

类型

- - X

文本文档

修改日期

2015/6/7 22:18

搜索

public class RandomAccessFileDemo{ public static void main(String[] args) throws IOException { writeFile(); 07. } 08. //使用RandomAccessFile对象写入一些人员信息,比如姓名和年龄 09. public static void writeFile() throws IOException { 10. //如果文件不存在,则创建,如果文件存在,不创建 11. RandomAccessFile raf = new RandomAccessFile("ranacc.txt" ,"rw"); raf.write("张三".getBytes()); 14 //使用write方法之写入最后一个字节 15. raf.write(97); 16. //使用writeInt方法写入四个字节(int类型) 17. 18. raf.writeInt(97); 19. raf.write("小强".getBytes()); 20. raf.writeInt(99); 21. 22. 23. raf.close(); 24. } 25. 复制代码 运行结果: _ 0 X ■ 管理员: C:\Windows\system32\cmd.exe

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H) 张三a a小强 c 示例2: 01. import java.io.IOException; 02. import java.io.RandomAccessFile; 03. 04. public class RandomAccessFileDemo{ public static void main(String[] args) throws IOException { readFile(); 06. 07. } 08. public static void readFile() throws IOException { 09. RandomAccessFile raf = new RandomAccessFile("ranacc.txt" ,"r"); 10. 11. //通过seek设置指针的位置 12. raf.seek(9); //随机的读取,只要指定指针的位置即可 13. 14. byte[] buf = new byte[4]; 15. raf.read(buf); 16. 17. String name = new String(buf); 18. 19. System.out.println("name=" + name); 21. int age = raf.readInt(); System.out.println("age=" + age); 22. 23. System.out.println("pos:" + raf.getFilePointer()); 24. 25. raf.close(); 27. } 28. } 复制代码

```
- - X
■ 管理员: C:\Windows\system32\cmd.exe
D:\code\day24>javac RandomAccessFileDemo.java
D:\code\day24>java RandomAccessFileDemo
name=小强
age=99
pos:17
示例3:
01.
      import java.io.IOException;
      import java.io.RandomAccessFile;
02.
03.
      public class RandomAccessFileDemo{
04.
             public static void main(String[] args) throws IOException {
05.
                   randomWrite();
07.
            }
08.
             public static void randomWrite() throws IOException {
09.
                   RandomAccessFile raf = new RandomAccessFile("ranacc.txt" ,"rw" );
10.
11.
                   //往指定位置写入数据
12.
                   raf.seek(3*8);
14.
                   raf.write("哈哈".getBytes());
15.
                   raf.writeInt(102);
16.
17.
18.
                  raf.close();
19.
            }
20.
      复制代码
 运行结果:
                                                                         _ D X
■ 管理员: C:\Windows\system32\cmd.exe
D:∖code∖day24>javac RandomAccessFileDemo.java
D:\code\day24>java RandomAccessFileDemo
             ▶ 计算机 ▶ 本地磁盘 (D:) ▶ code ▶ day24
 文件(F) 编辑(E) 查看(V)
                        工具(T) 帮助(H)
  组织▼
            ∭ 打开 ▼
                         新建文件夹
                                                              修改日期
                         名称
                                                                               类型
   ☆ 收藏夹
    🏮 下载
                         anacc.txt
                                                              2015/6/7 22:21
                                                                               文本文档
                                                                         🧻 ranacc.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
张三a a小强 c
                    哈哈 f
 管道流
 PipedInputStream和PipedOutputStream:输入输出可以直接进行连接,通过结合线程使用。
示例1:
      import java.io.PipedInputStream;
01.
      import java.io.PipedOutputStream;
02.
03.
      public class PipedStream{
04.
          public static void main(String[] args) throws Exception {
05.
06.
             PipedInputStream input = new PipedInputStream();
07.
             PipedOutputStream output = new PipedOutputStream();
08.
09.
             input.connect(output);
10.
11.
              new Thread(new Input(input)).start();
              new Thread(new Output(output)).start();
12.
13.
          }
14.
      }
15.
16.
      class Input implements Runnable{
17.
          private PipedInputStream in;
18.
          Input(PipedInputStream in){
19.
20.
              this.in = in;
          }
21.
22.
23.
          public void run(){
24.
              try{
                 byte[] buf = new byte[1024];
25.
                 int len = in.read(buf);
26.
27.
28.
                 String s = new String(buf,0,len);
                 System.out.println( "s=" + s);
29.
30.
                 in.close();
31.
             } catch(Exception e){
32.
                 e.printStackTrace();
             }
33.
          }
35.
      }
36.
37.
      class Output implements Runnable{
38.
          private PipedOutputStream out;
39.
          Output(PipedOutputStream out){
40.
              this.out = out;
41.
42.
          }
43.
          public void run(){
44.
45.
              try{
                 out.write( "hi,管道来了! " .getBytes());
46.
                 out.close();
47.
             } catch(Exception e){
48.
                 e.printStackTrace();
49.
             }
50.
51.
          }
52.
      复制代码
运行结果:
                                                                         - - X
■ 管理员: C:\Windows\system32\cmd.exe
D:\code\day24>javac PipedStream.java
D:\code\day24>java PipedStream
 =hi.管道来了!
操作基本数据类型
 DataInputStream与DataOutputStream
示例1:
01.
      import java.io.DataOutputStream;
      import java.io.FileOutputStream;
02.
03.
      import java.io.IOException;
04.
05.
      public class DataStreamDemo{
          public static void main(String[] args) throws IOException {
06.
             writeData();
07.
08.
          }
09.
           public static void writeData() throws IOException {
10.
             DataOutputStream dos = new DataOutputStream(new
11.
      FileOutputStream("data.txt"));
12.
13.
             dos.writeUTF( "您好");
14.
             dos.close();
15.
16.
        }
17.
     }
      复制代码
运行结果:
                                                                         - - X

■ 管理员: C:\Windows\system32\cmd.exe

D:\code\day24>javac DataStreamDemo.java
D:\code\day24>java DataStreamDemo
                                                                            + 44

        →

        →
        计算机
        →
        本地磁盘 (D:)
        →
        code
        →
        day24

        编辑(E) 查看(V) 工具(T) 帮助(H)
           打开▼
  组织▼
                        新建文件夹
                        名称
                                                            修改日期
                                                                            类型
  ☆ 收藏夹
                                                             2015/6/7 22:27
                         data.txt
                                                                             文本文档
                                                                          _ D X
🧻 data.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
 □您好
示例2:
      import java.io.DataInputStream;
01.
      import java.io.FileInputStream;
02.
      import java.io.IOException;
03.
04.
      public class DataStreamDemo{
             public static void main(String[] args) throws IOException {
06.
                  readData();
07.
            }
08.
09.
             public static void readData() throws IOException {
10.
                  DataInputStream dis = new DataInputStream(new
11.
      FileInputStream("data.txt" ));
12.
13.
                  String str = dis.readUTF();
14.
                  System.out.println(str);
15.
16.
                  dis.close();
17.
18.
            }
19.
      复制代码
运行结果:
                                                                         _ _ X

■ 管理员: C:\Windows\system32\cmd.exe

D:\code\day24>javac DataStreamDemo.java
D:\code\day24>java DataStreamDemo
操作字节数组
 ByteArrayInputStream与ByteArrayOutputStream
P.S.
 关闭字节数组输出输出流无效,因为它们没有调用底层资源,所有的操作都是在内存中完成的。
示例1:
      import java.io.ByteArrayInputStream;
01.
02.
      import java.io.ByteArrayOutputStream;
      import java.io.IOException;
03.
04.
05.
     public class ByteArrayStreamDemo{
             public static void main(String[] args) throws IOException {
06.
                   ByteArrayInputStream bis = new ByteArrayInputStream("abcdef"
      .getBytes());
08.
                   ByteArrayOutputStream bos = new ByteArrayOutputStream();
09.
10.
                   int ch = 0;
11.
                   while((ch = bis.read()) != -1){
12.
                         bos.write(ch);
13.
14.
15.
                   System.out.println(bos.toString());
17.
            }
      }
18.
      复制代码
```

```
15.
16.
17.
           }
18.
            private static void printBytes(byte[] buf){
19.
```

运行结果:

abcdef

8.2.12 编码表

编码表的由来

常见的编码表

示例1:

01.

02.

03.

04.

05.

06.

07.

09. 10.

11. 12.

13.

15.

16. 17.

18. 19.

20. 21.

22. 23. 24.

25.

26. 27. 28.

29.

30.

}

运行结果:

-60 -6 -70 -61

示例2:

01.

02.

03.

04.

05.

06.

07.

08.

09.

10.

11.

12.

13.

14.

20.

21.

22.

23.

24.

复制代码

}

}

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code\day24>java EncodeDemo

-26 -126 -88 -27 -91 -67

如果编码编错了,解不出来。

如果编对了,解错了,有可能有救。

import java.io.IOException;

public class EncodeDemo{

String str = "您好";

byte[] buf = str.getBytes("gbk");

System.out.println("s1 = " + s1);

String s1 = new String(buf,"iso8859-1");

byte[] buf2 = s1.getBytes("iso8859-1");

String s2 = new String(buf2, "gbk");

System.out.println(s2);

for(byte b : buf){

}

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day24>java EncodeDemo

如果编对了,解错了,也可能没救了。

import java.io.IOException;

String str = "您好";

printBytes(buf2);

System.out.println(s2);

for(byte b : buf){

System.out.println();

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

byte[] buf = str.getBytes("gbk");

String s1 = new String(buf, "UTF-8");

byte[] buf2 = s1.getBytes("UTF-8");

String s2 = new String(buf2, "gbk");

private static void printBytes(byte[] buf){

"您好"的gbk编码在UTF-8码表中查不到对应的字符,所以已经用"?"代替。

故即使使用UTF-8码表进行解码,获取的字节也不是"您好"的gbk编码后的字节。

因此,使用UTF-8码表对 " $\pi\pi$ " 进行解码,获取的字节也依然是 "您好" 的gbk编码后的字节。

"谢谢"的gbk编码在UTF-8码表中可以查到对应的字符,为" $\pi\pi$ "。

🕌 ▶ 计算机 ▶ 本地磁盘 (D:) ▶ code ▶ day24

工具(T) 帮助(H)

System.out.print(b + " ");

System.out.println("s1 = " + s1);

public class EncodeDemo{

D:∖code∖day24>javac EncodeDemo.java

}

复制代码

}

运行结果:

s1 = ?u??

示例3:

01.

02. 03.

06.

07. 08.

09. 10. 11.

13. 14.

15. 16.

17. 18.

20.

21.

22.

23.

24. 25.

27. 28.

}

运行结果:

原因分析:

P.S.

复制代码

}

}

配 管理员: C:\Windows\system32\cmd.exe

D:\code\day24>java EncodeDemo

17 -65 -67 -17 -65 -67

所以再也不能成功解码了。

实验:联通乱码问题。

1. 新建一个1.txt文件。

文件(F) 编辑(E) 查看(V)

10.

13.

15.

16.

17.

18.

19.

20.

21.

22.

23.

弃

cutStringByByte(str,x+1));

int count = 0;

if(buf[x] < 0)

else

count++;

break;

byte[] buf = str.getBytes("gbk");

for(int x = len - 1; x > = 0; $x - -){$

}

D:∖code∖day24>javac EncodeDemo.java

"?"在UTF-8中的编码为-17 -65 -67

所以,不会出现"您好"发生的情况。

System.out.println(b);

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

D:∖code∖day24>javac EncodeDemo.java

GB2312:中国的中文编码表。

Unicode:国际标准码,融合了多种文字。

UTF-8:最多用三个字节来表示一个字符。

import java.io.IOException;

//编码

//解码

public class EncodeDemo{

字。

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day24>javac ByteArrayStreamDemo.java

计算机只能识别二进制数据,早期由来是电信号。为了方便应用计算机,让它可以识别各个国家的文

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

就将各个国家的文字用数字来表示,并一一对应,形成一张表,这就是编码表。

ASCII:美国标准信息交换码,用一个字节的7位可以表示。

ISO8859-1:拉丁码表。欧洲码表,用一个字节的8位表示。

GBK:中国的中文编码表升级,融合了更多的中文文字符号。

所有文字都用两个字节来表示,Java语言使用的就是unicode

//字符串-->字节数组:编码

//字符数组-->字符串:解码

byte[] buf1 = str.getBytes("GBK");

byte[] buf2 = str.getBytes("UTF-8");

String s2 = new String(buf2,"UTF-8");

System.out.print(b + " ");

System.out.println("s2 = " + s2);

private static void printBytes(byte[] buf){

for(byte b : buf){

System.out.println();

String s1 = new String(buf1); System.out.println("s1 = " + s1);

String str = "您好";

printBytes(buf1);

printBytes(buf2);

D:∖code∖day24>java ByteArrayStreamDemo

_ D X

_ D X

_ D X

- - X

¥ 44

```
打开▼
 组织▼
                   打印
                         新建文件夹
                   名称
                                                修改日期
  ☆ 收藏夹
                   1.txt
                                                2015/6/7 22:39
                                                             文本文档
2. 输入联通,保存。
🧻 1.txt - 记事本
文件(E) 编辑(E) 格式(O) 查看(V) 帮助(H)
联通
3. 关闭,重新打开此文件,发现乱码。
🧻 1.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
原因分析:
 "联通"经过gbk编码后成四个字节:11000001、10101010、11001101、10101000。
正好符合UTF-8的编码规则。所以,记事本按照UTF-8进行了解码,从而出现了乱码现象。
练习:
在java中,字符串 "abcd"与字符串 "ab您好"的长度是一样,都是四个字符。
但对应的字节数不同,一个汉字占两个字节。
定义一个方法,按照最大的字节数来取子串。
如:对于"ab你好",如果取三个字节,那么子串就是ab与"你"字的半个。
那么半个就要舍弃。如果取四个字节就是 "ab你" , 取五个字节还是 "ab你" 。
代码:
    import java.io.IOException;
01.
02.
    public class Test{
03.
        public static void main(String[] args) throws IOException {
          String str = "ab你好cd谢谢";
05.
06.
          int len = str.getBytes("gbk" ).length;
07.
08.
          for(int x = 1; x < len; x++){
09.
```

System.out.println("截取的" + (x + 1) + "个节结果是:" +

public static String cutStringByByte(String str,int len) throws IOException {

//gbk编码的值两个字节值一般都为负,记录连续的负数个数,如果为奇数,则舍

} 24. 25. if(count % 2 == 0){ 26. return new String(buf,0,len,"gbk"); 27. } else{ return new String(buf,0,len-1,"gbk"); 29. } 30. 31. } } 32. 复制代码 运行结果: - - X ■ 管理员: C:\Windows\system32\cmd.exe D:∖code∖day24>javac Test.java D:\code\day24>java Test P.S. 中文经过gbk编码后,也有两个字节不都为负数的情况,例如"琲",字节值为-84、105。

第一个字节值为负,第二个字节值为正。因此,上面的代码得出的结果依然正确。

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

System.out.println("截取的" + (x + 1) + "个节结果是: " +

int len = str.getBytes("utf-8").length;

定义一个方法,按照最大的字节数来取子串,使用UTF-8编码的代码如下:

String str = "ab你好cd谢谢";

for(int x = 1; x < len; x++){

01. public class Test{

}

cutStringByByte(str,x+1));

}

03. 04.

05.

07.

08.

09.

10.

