

```
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

8.2.7 File类

需求:获取c盘目录下的隐藏文件。

public class FileListDemo{

03.

04.

代码: import java.io.File; 01.

public static void main(String[] args){

import java.io.FilenameFilter; 02.

listDemo(); 06. 07. } 08. 09. public static void listDemo(){ 10. File dir = new File("c:\\"); 12. File[] files = dir.listFiles(new FilterByHidden()); 13. for(File file : files){ 14. System.out.println(file); 15. } 16.

17. } 18. } 19. 20. class FilterByHidden implements FilenameFilter{ public boolean accept(File dir,String name){ 21. 22. return dir.isHidden(); } 23. 24. 复制代码 运行结果: - 0 X 画 管理员: C:\Windows\system32\cmd.exe D:\code>cd day23 D:\code\day23>javac FileListDemo.java D:\code\day23>java FileListDemo

c:\\$Recycle.Bin c:\Boot c:\bootmgr c: Documents and Settings c:\Intel c:\myclass c:\OEMSF ::\pagefile.sys c:\PerfLogs : Program Files :\Program Files (x86) : ProgramData c:\System Volume Information : Wsers ::win7 激活工具 :: Windows 8.2.8 递归 函数自身直接或者间接的调用到了自身。 一个功能在被重复使用,并每次使用时,参与运算的结果和上一次调用有关。这时可以用递归来解决 问题。

H

P.S. 1、递归一定明确条件,否则容易栈溢出。 2、注意一下递归的次数。 需求:对指定目录进行所有内容的列出(包含子目录中的内容),也可以理解为深度遍历。 代码: 01. import java.io.File; public class FileListDemo{ 03. public static void main(String[] args){ 04.

listAll(dir,0); 06. 07. 08. 09. public static void listAll(File dir, int level){ 10. System.out.println(getSpace(level) + "dir:" + dir.getAbsolutePath()); //获取指定目录下当前的所有文件夹或者文件对象 11. level++; 12. File[] files = dir.listFiles(); 13.

05.

14.

15.

23.

24. 25.

26. 27.

28. 29.

30. 31.

32.

33.

运行结果:

运行结果:

代码:

01.

02.

03.

04.

05.

06.

07.

08.

09.

10.

代码:

01.

02.

03.

04.

05.

07.

08.

09.

10.

11.

12.

13.

}

import java.io.File;

public class RemoveDirTest{

removeDir(dir);

public static void main(String[] args){

File dir = new File("d:\\demo");

public static void removeDir(File dir){

if(file.isDirectory()){

File[] files = dir.listFiles();

for(File file : files){

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day23>java RemoveDirTest 鋫

d:\demo\1 -

d:∖demo∖1 -

d:∖demo\1 -

d: \demo \1

d:\demo\1

d:\demo\1

Map

特点:

示例1:

22.

23.

24.

25.

26.

27.

28.

29.

30.

运行结果:

zhangsan:10 lisi:20 wangwu:26 zhaoliu:40

示例2:

01.

02. 03.

05.

06.

07. 08.

09.

10.

11.

12. 13.

14. 15. 16.

17.

18.

运行结果:

zhangsan=10 lisi=20 wangwu=30 zhaoliu=40

示例3:

01.

02. 03.

04. 05.

06.

07.

08.

09. 10.

11.

12. 13.

14.

示例4:

import java.io.BufferedReader;

import java.io.FileInputStream;

import java.io.FileReader;

import java.io.FileWriter; import java.io.IOException; import java.util.Properties;

public class PropertiesDemo{

}

propertiesDemo();

//需要使用到读取流

//使用load方法

prop.load(fis);

test();

myLoad();

prop.list(System.out);

//对已有的配置文件中的信息进行修改。

//再通过流将修改后的数据存储到文件中

if(!file.exists()){

//创建集合存储配置信息

//将流中信息存储到集合中

prop.load(fr);

prop.store(fw, "");

fr.close();

}

D:∖code∖day23>java PropertiesDemo

import java.io.FileOutputStream;

import java.io.IOException;

import java.util.Properties;

public class PropertiesTest{

getAppCount();

//将配置文件封装成File对象

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

public static void getAppCount() throws IOException {

File confile = new File("count.properties");

listing properties

listing properties --

zhangsan=10 lisi=20 zhaoliu=40 wangwu=30

zhangsan=10 lisi=20 wangwu=26 haoliu=40

程序。

算。

03.

04.

05.

07.

08.

09.

10.

11.

12.

13.

14.

}

//读取这个文件

public static void test() throws Exception {

File file = new File("info.txt");

file.createNewFile();

Properties prop = new Properties();

prop.setProperty("wangwu","26");

FileWriter fw = new FileWriter(file);

FileReader fr = new FileReader("info.txt");

import java.io.File;

01.

02.

03.

04. 05.

07. 08.

09.

10.

11. 12.

14

15.

16.

17.

18.

19. 20.

21.

22.

23.

24. 25. 26.

27.

28. 29.

30.

31.

32.

33.

34. 35.

36. 37.

38.

39. 40.

41.

43.

44. 45.

46.

47. 48.

50.

51. 52.

53. 54. }

👞 管理员: C:\Windows\system32\cmd.exe

D:\code\day23>java PropertiesDemo

演示Properties集合和流对象相结合的功能。

import java.util.Properties;

public class PropertiesDemo{

}

}

画 管理员: C:\Windows\system32\cmd.exe

D:\code\day23>java PropertiesDemo

listing properties

}

D:∖code∖day23>javac PropertiesDemo.java

import java.io.FileOutputStream;

propertiesDemo();

import java.util.Properties;

public class PropertiesDemo{

复制代码

public static void main(String[] args){

public static void propertiesDemo(){

prop.list(System.out);

Properties prop = new Properties();

prop.setProperty("zhangsan","10");

prop.setProperty("lisi", "20");

prop.setProperty("wangwu","30"); prop.setProperty("zhaoliu","40");

propertiesDemo();

D:\code\day23>javac PropertiesDemo.java

}

复制代码

//取出所有元素

for(String name : names){

d:\demo\1.txt:true d∶∖demo∖demo2√2 -

d:\demo\demo2\2

d:\demo\demo2\2

d:\demo\demo2\2

d:\demo\demo2\2

8.2.9 Properties集合

|--Hashtable

|--Properties

Properties集合的存和取。

1. 该集合中的键和值都是字符串类型。

2. 集合中的数据可以保存到流中,或者从流中获取。

3. 通常该集合用于操作以键值对形式存在的配置文件。

D:∖code∖day23>javac RemoveDirTest.java

(2).txt:true

(3).txt:true

(4).txt:true (5).txt:true

(6).txt:true

(7).txt:true (8).txt:true

(9).txt:true .txt:true

(2).txt:true

(3).txt:true

(4).txt:true

(5).txt:true (6).txt:true

}

■ 管理员: C:\Windows\system32\cmd.exe

D:\code\day23>java DiGuiDemo

需求:利用递归求1到10的和。

D:\code\day23>javac DiGuiDemo.java

public class DiGuiDemo{

public static void main(String[] args){

int sum = getSum(10);

System.out.println(sum);

public static int getSum(int num){

return num + getSum(num - 1);

if(num == 1)

return 1;

File dir = new File("D:\\Java\\jdk1.6.0_02\\include");

if (files[x].isDirectory()){ 16. 17. listAll(files[x],level); 18. System.out.println(getSpace(level) + "file:" + 19. files[x].getAbsolutePath()); 20. } 21. } 22.

private static String getSpace(int level){

for (int x = 0; x < level; x++){ sb.append("| ");

sb.append("|--");

return sb.toString();

}

}

画 管理员: C:\Windows\system32\cmd.exe

D:\code\day23>java FileListDemo -dir:D:\Java\jdk1.6.0_21\include

D:\code\day23>javac FileListDemo.java

复制代码

StringBuilder sb = new StringBuilder();

_ 0 X

_ D X

III

for (int x = 0; $x < files.length; x++){$

-¦ file:D:\Java\jdk1.6.0_21\include\classfile_constants.h -{ file:D:\Java\jdk1.6.0_21\include\jawt.h -¦ file:D:\Java\jdk1.6.0_21\include\jdwpTransport.h -¦ file:D:\Java\jdk1.6.0_21\include\jni.h -¦ file:D:\Java\jdk1.6.0_21\include\jvmti.h -¦ dir:D:\Java\jdk1.6.0_21\include\win32 -¦ ¦ file:D:\Java\jdk1.6.0_21\include\win32\jawt_md.h $\label{line:definition} \begin{tabular}{ll} file:D: \Java idk1.6.0_21 include \win32 ini_md.h. \end{tabular}$ file:D: Java jdk1.6.0_21 \include \win32 需求:利用递归求6的二进制值。 代码: 01. public class DiGuiDemo{ 02. public static void main(String[] args){ 03. toBin(6); 04. } 05. public static void toBin(int num){ 06. if(num > 0){ 07. toBin(num/2); 09. System.out.print(num%2); 10. 11. } 12. 复制代码

11. } 12. 复制代码 运行结果: _ D X ■ 管理员: C:\Windows\system32\cmd.exe D:\code\day23>javac DiGuiDemo.java 111 D:∖code∖day23>java DiGuiDemo 55 需求:删除一个带内容的目录。 原理:必须从里面往外删,需要进行深度遍历。

removeDir(file); 14. 15. } else{ System.out.println(file + ":" + file.delete()); 16. 17. 18. } System.out.println(dir + ":" + dir.delete()); 19. 20. } 21. } 复制代码 运行结果:

_ D X

import java.util.Properties; 01. import java.util.Set; 03. public class PropertiesDemo{ 04. public static void main(String[] args){ 05. propertiesDemo(); 06. } 07. 08. public static void propertiesDemo(){ 09. //创建一个Properties集合 10. Properties prop = new Properties(); 11. 12. //存储元素 13. prop.setProperty("zhangsan","10"); 14. prop.setProperty("lisi","20"); 15. 16. prop.setProperty("wangwu","30"); prop.setProperty("zhaoliu","40"); 17. 18. 19. //修改元素 20. prop.setProperty("wangwu","26"); 21.

Set < String > names = prop.stringPropertyNames();

String value = prop.getProperty(name);

System.out.println(name + ":" + value);

- - X

_ D X

111

prop.setProperty("zhaoliu","40"); 15. 16. //想要将这些集合中的字符串键值信息持久化存储到文件中 17. //需要关联输出流 18. FileOutputStream fos = new FileOutputStream("info.txt"); 19. 20. //将集合中数据存储到文件中 21. 22. prop.store(fos, "name+age"); 23. 24. fos.close(); 25. } 26. 复制代码 运行结果: - - X ■ 管理员: C:\Windows\system32\cmd.exe D:∖code∖day23>javac PropertiesDemo.java D:\code\day23>java PropertiesDemo 計算机 → 本地磁盘 (D:) → code → day23 v 44 文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H) 新建文件夹 组织▼ 包含到库中 ▼ 共享▼ 名称 修改日期 类型 ☆ 收藏夹 info.txt 2015/6/7 19:33 🥘 info.txt - 记事本 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H) #name+age #Sun Jun 07 19:33:07 CST 2015 zhangsan=10 lisi=20 zhaoliu=40 wangwu=30

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

public static void propertiesDemo() throws Exception {

FileInputStream fis = new FileInputStream("info.txt");

//读取这个文件。并将这个文件中的键值数据存储到集合中。再通过集合对数据进行修改。

Properties prop = new Properties();

//注意:必须要保证该文件中的数据是键值对。

//集合中的数据来自于一个文件。

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

public static void propertiesDemo() throws Exception {

Properties prop = new Properties();

prop.setProperty("zhangsan","10");

prop.setProperty("lisi","20"); prop.setProperty("wangwu","30");

//模拟一下load方法 public static void myLoad() throws Exception { 57. 58. Properties prop = new Properties(); 59. BufferedReader bufr = new BufferedReader(new 60. FileReader("info.txt")); 61. String line = null; 63. while((line = bufr.readLine()) != null){ 64. if(line.startsWith("#")) 65. 66. continue; String[] arr = line.split("="); 67. prop.setProperty(arr[0],arr[1]); 68. } 69. 70. 71. prop.list(System.out); bufr.close(); 72. } 73. 74. 复制代码 运行结果: _ D X 画 管理员: C:\Windows\system32\cmd.exe D:\code\day23>javac PropertiesDemo.java

E

思路: 1、应该有计数器 每次程序启动都需要计数一次,并且是在原有的次数上进行计数。 2、计数器就是一个变量。突然冒出一想法,程序启动时进行计数,计数器必须存在于内存并进行运 可是程序一结束, 计数器消失了。那么再次启动该程序, 计数器又重新被初始化了。 而我们需要多次启动同一个应用程序,使用的是同一个计数器。 这就需要计数器的生命周期边长,从内存存储到硬盘文件中。 3、如何使用这个计数器呢? 首先,程序启动时,应该先读取这个用于记录计数器信息的配置文件。 获取上一次计数器次数。并进行使用次数的判断。 其次,对该次数进行自增,并自增后的次数重新存储到配置文件中。 4、文件中的信息该如何进行存储并体现。 直接存储次数值可以,但是不明确该数据的含义。所以起名字就变得很重要。 这就有了名字和值的对应,所以可以使用键值对。 可是映射关系map集合搞定,又需要读取硬盘上的数据,所以map+io=properties。 代码: import java.io.File; 01. import java.io.FileInputStream; 02.

需求:获取一个应用程序运行的次数,如果超过5次,给出使用次数已到请注册的提示,并不要再运行

```
15.
             if(!confile.exists()){
 16.
                confile.createNewFile();
 17.
 18.
             }
 19.
             FileInputStream fis = new FileInputStream(confile);
 21.
             Properties prop = new Properties();
 22.
 23.
 24.
             prop.load(fis);
 25.
             //从集合中通过键获取次数
 26.
 27.
             String value = prop.getProperty( "time");
 28.
 29.
             //定义计数器,记录获取到的次数
 30.
             int count = 0;
 31.
              if(value != null){
 32.
                count = Integer.parseInt(value);
 33.
                 if(count > = 5){
 34.
                    throw new RuntimeException("使用次数已到,请注册,给钱!");
 35.
                }
 36.
             }
 37.
             count++;
 38.
 39.
             //将改变后的次数重新存储到集合中。
 40.
             prop.setProperty( "time",count + "" );
 41.
             FileOutputStream fos = new FileOutputStream(confile);
 42.
             prop.store(fos, "");
 44.
 45.
 46.
             fos.close();
 47.
             fis.close();
 48.
          }
 49.
      }
       复制代码
 运行结果:
                                                                   _ _ _ X
 画 管理员: C:\Windows\system32\cmd.exe
 D:\code\day23>javac PropertiesTest.java
D:\code\day23>java PropertiesTest
                                                                              E
D:\code\day23>java PropertiesTest
D:\code\day23>java PropertiesTest
 D:\code\day23>java PropertiesTest
D:\code\day23>java PropertiesTest
 D:\code\day23>java PropertiesTest
 Exception in thread "main" java.lang.RuntimeException: 使用次数已到,请注册,给
 钱!
        at PropertiesTest.getAppCount(PropertiesTest.java:34)
        at PropertiesTest.main(PropertiesTest.java:9)
  需求:获取指定目录下,指定扩展名的文件(包含子目录中的),并且将这些文件的绝对路径写入到
一个文本文件中。
 简单说:就是建立一个指定扩展名的文件的列表。
 思路:
 1. 必须进行深度遍历。
  2. 要在遍历的过程中进行过滤,将符合条件的内容都存储到容器中。
 3. 对容器中的内容进行遍历并将绝对路径写入到文件中。
 代码:
 01.
       import java.io.BufferedWriter;
       import java.io.File;
 03.
      import java.io.FileWriter;
 04.
      import java.io.FilenameFilter;
      import java.io.IOException;
 05.
      import java.util.ArrayList;
 06.
 07.
      import java.util.List;
 08.
 09.
       public class Test{
 10.
             public static void main(String[] args) throws IOException {
                  File dir = new File("d:\\code" );
 11.
 12.
                  FilenameFilter filter = new FilenameFilter(){
 13.
                        public boolean accept(File dir,String name){
                              return name.endsWith(".java" );
 14.
 15.
                        }
                  };
 17.
 18.
                  List<File> list = new ArrayList<File>();
 19.
                  getFiles(dir,filter,list);
 20.
                  File destFile = new File(dir, "javalist.txt" );
                  write2File(list,destFile);
 21.
 22.
            }
            对指定目录中的内容进行深度遍历,并按照指定过滤器,进行过滤。
            将过滤后的内容存储到指定容器List中。
 25.
 26.
 27.
             public static void getFiles(File dir,FilenameFilter filter,List<File>
       list){
 28.
                  File[] files = dir.listFiles();
 29.
                  for(File file : files){
 30.
                        //递归
 31.
                        if(file.isDirectory()){
 32.
                             getFiles(file,filter,list);
 33.
 34.
                        } else{
                              //对便利到的文件进行过滤器的过滤。将符合条件File对象,存储到
 35.
       List集合中
 36.
                              if(filter.accept(dir,file.getName())){
 37.
                                  list.add(file);
 38.
                             }
 39.
                       }
 40.
                  }
 41.
            }
 42.
             public static void write2File(List<File> list,File destFile) throws
       IOException{
 44.
                  BufferedWriter bufw = null;
                  try{
 45.
                        bufw = new BufferedWriter(new FileWriter(destFile));
 46.
 47.
                        for(File file : list){
 49.
                             bufw.write(file.getAbsolutePath());
 50.
                             bufw.newLine();
                             bufw.flush();
 51.
 52.
                        }
 53.
                  } finally{
                        if(bufw!=null)
 54.
                              try{
                                   bufw.close();
 56.
 57.
                             } catch(IOException e){
                                   throw new RuntimeException("关闭失败");
 58.
 59.
                             }
 60.
                  }
 61.
            }
 62.
      复制代码
 运行结果:
                                                                   _ D X
 🚾 管理员: C:\Windows\system32\cmd.exe
 D:\code\day23>javac Test.java
 D:\code\day23>java Test

    → → 计算机 → 本地磁盘 (D:) → code →
                                                                      + 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
                       名称
                                                        修改日期
                                                                       类型
   ☆ 收藏夹
                       javalist.txt
                                                        2015/6/7 19:41
                                                                       文本文档
                                                                   🥘 javalist.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
 d:\code\day23\PropertiesDemo.java
 d:\code\day23\PropertiesTest.java
 d:\code\day23\RemoveDirTest.java
 d:\code\day23\Test.java
 8.2.10 IO包中的其他类
 打印流
  PrintWriter与PrintStream:可以直接操作输入流和文件。
  PrintStream为其他输出流添加了功能,使它们能够方便地打印各种数据值表示形式。
  与其他输出流不同, PrintStream永远不会抛出IOException。
  PrintStream打印的所有字符都使用平台的默认字符编码转换为字节。
  在需要写入字符而不是写入字节的情况下,应该使用PrintWriter类。
  PrintStream:
 1. 提供了打印方法可以对多种数据类型值进行打印,并保持数据的表示形式
```

III print.txt - 记事本 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H) a97 PrintWriter:字符打印流

构造函数参数:

1. 字符串路径

3. 字节输出流

4. 字符输出流

示例2:

01.

02.

03.

04.

06.

07.

08.

09. 10.

11.

12.

13.

14.

05. 06.

07.

08. 09.

10.

11. 12.

13.

14. 15.

16.

17.

18.

19.

20. 21.

22. 23.

24.

运行结果:

abcde fgklm over

文件(F)

组织

☆ 收藏夹 🏮 下载

■ out.txt - 记事本

ABCDE **FGKLM**

序列流

22. 23.

25.

26. 27.

28.

29. 30.

31.

32.

运行结果:

组织▼

■ 图片

1 文档

音乐

1.txt - 记事本

🥘 2.txt - 记事本

🥘 3.txt - 记事本

图片

▶ 🖹 文档

音乐

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.

23.

24.

25.

26.

27.

28.

29.

30.

31.

32.

33.

34.

35.

45.

46.

47.

48.

}

复制代码

}

}

}

sis.close();

};*/

▷ 🗐 迅雷下载

abcde

fgklm

nopqr

📄 迅雷下载

编辑(E)

),true);

2. File对象

2. 它不抛IOException

1. 字符串路径

3. 字节输出流

2. File对象

示例1:

01.

02.

03.

04.

05.

06.

07.

08.

09.

10.

11.

12.

13.

14.

}

复制代码

}

运行结果:

组织▼

🛮 🥎 收藏夹 🎍 下载

构造函数,接收三种类型的值:

import java.io.PrintStream;

public class PrintStreamDemo{

out.write(97); //a

out.print(97); //97

out.close();

■ 管理员: C:\Windows\system32\cmd.exe

D:∖code∖day23>java PrintStreamDemo

文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

打开▼

D:\code\day23>javac PrintStreamDemo.java

🥌 ▶ 计算机 ▶ 本地磁盘 (D:) ▶ code ▶ day23

新建文件夹

print.txt

名称

import java.io.BufferedReader; import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.io.IOException;

public class PrintWriterDemo{

InputStreamReader(System.in));

import java.io.PrintWriter; import java.io.IOException;

public class PrintWriterDemo{

InputStreamReader(System.in));

}

}

■ 管理员: C:\Windows\system32\cmd.exe

D:\code\day23>java PrintWriterDemo

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

SequenceInputStream:对多个流进行合并。

需求:将1.txt、2.txt、3、txt文件中的数据合并到一个文件中。

int len = 0;

fos.close();

sis.close();

→ → 计算机 → 本地磁盘 (D:) → code → day23

共享▼

名称

1.txt

2.txt

3.txt

工具(T) 帮助(H)

新建文件夹

修改日期

2015/6/7 20:10

2015/6/7 20:13

2015/6/7 20:11

类型

文本文档

文本文档

文本文档

0

0

- 44

类型

文本文档

文本文档

文本文档

文本文档

修改日期

2015/6/7 20:10

2015/6/7 20:13

2015/6/7 20:11

2015/6/7 20:14

搜罗

}

}

复制代码

文件(F) 编辑(E) 查看(V)

包含到库中 ▼

文件(F) 編辑(E) 格式(O) 查看(V) 帮助(H)

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

👞 管理员: C:\Windows\system32\cmd.exe

D:\code\day23>javac SequenceInputStreamDemo.java

→ 计算机 → 本地磁盘 (D:) → code → day23

新建文件夹

共享▼

名称

1.txt

2.txt

3.txt 4.txt

D:\code\day23>java SequenceInputStreamDemo

文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

包含到库中 ▼

while((len = sis.read(buf)) != -1){

fos.write(buf,0,len);

D:\code\day23>javac PrintWriterDemo.java

▶ 计算机 ▶ 本地磁盘 (D:) ▶ code ▶ day23

新建又件夹

查看(V) 工具(T) 帮助(H)

名称

out.txt

复制代码

out.close(); bufr.close();

String line = null;

//写入到out.txt文件中

String line = null;

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

PrintStream out = new PrintStream("print.txt");

//print方法将97先变成字符串保持原样将数据打印到目的地

- - X

¥ 44

类型

文本文档

修改日期

2015/6/7 20:01

搜索de

//write(int b)方法只写最低8位

```
break;
                        out.println(line.toUpperCase());
15.
16.
                        out.flush();
17.
                  }
18.
                  out.close();
19.
20.
                  bufr.close();
            }
21.
22.
      复制代码
运行结果:
                                                                       - - X
画 管理员: C:\Windows\system32\cmd.exe
D:\code\day23>javac PrintWriterDemo.java
D:\code\day23>java PrintWriterDemo
abcde
ABCDE
fgklm
FGKLM
over
示例3:
      import java.io.BufferedReader;
01.
02.
      import java.io.FileWriter;
03.
      import java.io.InputStreamReader;
```

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

//PrintWriter构造函数的第二个参数设置为true,表示自动刷新

PrintWriter out = new PrintWriter(new FileWriter("out.txt"

- - X

+ 44

类型

- - X

文本文档

修改日期

2015/6/7 20:06

BufferedReader bufr = new BufferedReader(new

while((line = bufr.readLine()) != null){

out.println(line.toUpperCase());

if("over" .equals(line))

break;

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

BufferedReader bufr = new BufferedReader(new

PrintWriter out = new PrintWriter(System.out);

while((line = bufr.readLine()) != null){

if("over" .equals(line))

```
代码:
01.
      import java.io.FileInputStream;
02.
     import java.io.FileOutputStream;
      import java.io.SequenceInputStream;
04.
     import java.util.Enumeration;
      import java.util.Vector;
05.
06.
     public class SequenceInputStreamDemo{
07.
08.
             public static void main(String[] args) throws Exception {
                  Vector<FileInputStream> v = new Vector<FileInputStream>();
09.
10.
                  v.add( new FileInputStream("1.txt" ));
12.
                  v.add( new FileInputStream("2.txt" ));
                  v.add( new FileInputStream("3.txt" ));
13.
14.
                  Enumeration<FileInputStream> en = v.elements();
15.
17.
                  SequenceInputStream sis = new SequenceInputStream(en);
18.
19.
                  FileOutputStream fos = new FileOutputStream("4.txt" );
20.
21.
                   byte[] buf = new byte[1024];
```

4.txt - 记事本 文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H) abcdefgklmnopgr 使用ArrayList比Vector效率更高。 改进后代码: import java.io.FileInputStream; 01. import java.io.FileOutputStream; 02. import java.io.SequenceInputStream; 04. import java.util.ArrayList; 05. import java.util.Collections; import java.util.Enumeration; 06. import java.util.Iterator; 07. 08. 09. public class SequenceInputStreamDemo{ public static void main(String[] args) throws Exception { 10. 11. 12. ArrayList < FileInputStream > al = new ArrayList < FileInputStream > (); for(int x = 1; x <= 3; x++){

al.add(new FileInputStream(x + ".txt"));

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

//Collections工具类的enumeration方法核心代码:

public boolean hasMoreElements(){

return it.hasMoreElements();

public FileInputStream nextElement(){

return it.next();

Enumeration < FileInputStream > en = Collections.enumeration(al);

Enumeration<FileInputStream> en = new Enumeration<FileInputStream>(){

36. 37. byte[] buf = new byte[1024]; 38. int len = 0; 39. 40. while((len = sis.read(buf)) != -1){ fos.write(buf,0,len); 42. } 43. fos.close();

SequenceInputStream sis = new SequenceInputStream(en);

FileOutputStream fos = new FileOutputStream("4.txt");