计算机科学技术学院
School of Computer Science and Technology

# Java 程序设计

## 第 10 章 输入、输出流

# 导读

## 主要内容

- File 类
- 文件字节输入、输出流
- 文件字符输入、输出流
- 缓冲流
- 随机流
- 数组流
- 数据流
- 对象流
- 序列化与对象克隆
- 使用 Scanner 解析文件
- 文件锁

# 概述

- 输入、输出流提供一条**通道**程序，可以使用这条通道读取源中的数据或把数据传送到目的地。
- 把输入流的指向称作源，程序从指向源的输入流中读取源中的数据；而输出流的指向是数据要去的一个目的地，程序通过向输出流中写入数据把数据传送到目的地

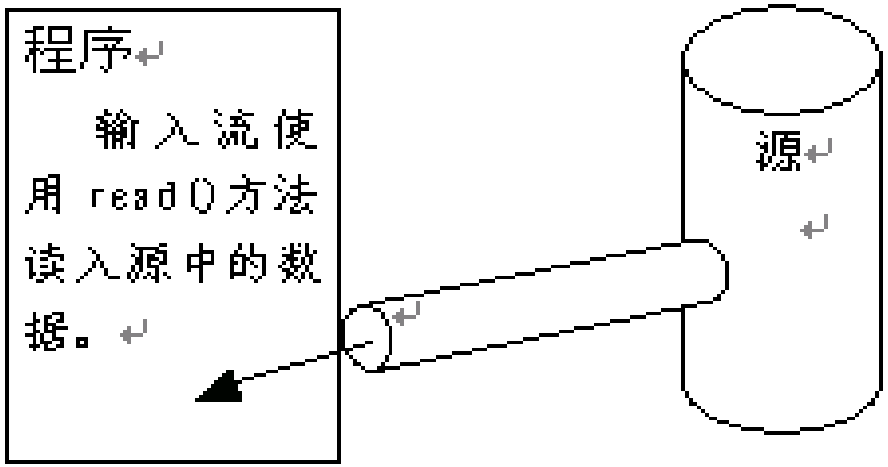


图 10.1 输入流示意图

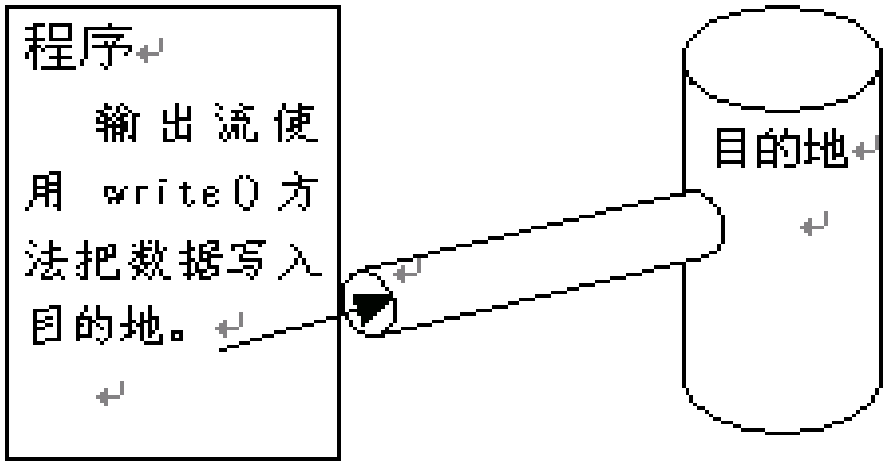


图 10.2 输出流示意图

# 10.1 File类

- File 对象主要用来获取文件本身的一些信息，不涉及对文件的读写操作。
- 创建一个 File 对象的构造方法有 3 个：
  - File(String filename);
  - File(String directoryPath, String filename);
  - File(File f, String filename);

# 10.1.1 文件的属性

- File 类的下列方法获取文件本身的一些信息。
  - public String getName() 获取文件的名字。
  - public boolean canRead() 判断文件是否是可读的。
  - public boolean canWrite() 判断文件是否可被写入。
  - public boolean exits() 判断文件是否存在。
  - public long length() 获取文件的长度（单位是字节）。
  - public String getAbsolutePath() 获取文件的绝对路径。
  - public String getParent() 获取文件的父目录。

# 10.1.1 文件的属性

- File 类的下列方法获取文件本身的一些信息。
  - public boolean isFile() 判断文件是否是一个普通文件，而不是目录。
  - public boolean isDirectroy() 判断文件是否是一个目录。
  - public boolean isHidden() 判断文件是否是隐藏文件。
  - public long lastModified() 获取文件最后修改的时间。

例子1( 例子中使用上述的一些方法，获取某些文件的信息 )

# 10.1.2 目录

1．创建目录 public boolean mkdir()

2．列出目录中的文件

- public String[] list() 用字符串形式返回目录下的全部文件。
- public String[] list(FilenameFilter obj) 指定类型。
- public File [] listFiles() 用 File 对象形式返回目录下的全部文件。
- public File [] listFiles(FilenameFilter obj) 指定类型。
- 上述两方法的参数 FilenameFilter 是一个接口，该接口有一个方法：
- public boolean accept(File dir, String name);

- 例子2 (Example10_2.java，FileAccept.java)，例子2 列出当前目录（应用程序所在的目录）下全部 java 文件的名字 )

# 10.1.3 文件的创建与删除

- 当使用 File 类创建一个文件对象后，例如
  - File file=new File("c:\\myletter","letter.txt");
  - 如果 c:\myletter 目录中没有名字为 letter.txt 文件，文件对象 file 调用方法 public boolean createNewFile();
  - 文件对象调用方法　public boolean delete() 可以删除当前文件，
  - 例如： file.delete();

# 10.1.4 运行可执行文件

- 用 Runtime 类声明一个对象 ( Runtime 类在 java.lang 包 )

  Runtime ec;

- 然后使用该类的 getRuntime() 静态方法创建这个对象：

  ec=Runtime.getRuntime();

  ec 可以调用 exec(String command) 方法打开本地机的可执行文件或执行一个操作。

例子3( 例子3中， Runtime 对象打开 windows 平台上的记事本程序和浏览器 )

- File file = new File("C:\\windows", “Notepad.exe");
- *rt.exec(file.getAbsolutePath());*

# 10.2 文件字节输入流

- java.io 包提供了大量的流类
- InputStream 抽象类的子类创建的流对象称作字节输入流；OutputStream 抽象类子类创建的流对象称作字节输出流
- Reader 抽象类的子类创建的流对象称作字符输入流； Writer 抽象类的子类创建的流对象称作字符输出流。
- 针对不同的源或目的地。 java.io 包为程序提供了相应的输入流或输出流。这些输入、输出流绝大部分都是 InputStream 、 OutputStream 、 Reader 或 Writer 的子类。

# 10.2 文件字节输入流

- 使用输入流通常包括 4 个基本步骤：

(1) 设定输入流的源

(2) 创建指向源的输入流

(3) 让输入流读取源中的数据

(4) 关闭输入流。

# 10.2 文件字节输入流

1. 构造方法：设定输入流源

使用 FileInputStream 类的下列构造方法创建指向文件的输入流。

FileInputStream(String name);

FileInputStream(File file);

其中：参数 name 和 file 指定的文件称为**输入流的源。**

# 10.2 文件字节输入流

## 2. 使用输入流读取字节

文件字节流可以调用从父类继承的 read 方法顺序地读取文件，只要不关闭流，每次调用 read 方法就顺序地读取文件中的其余内容，直到文件的末尾或文件字节输入流被关闭。

- **int read()** 读取单个字节的数据，返回字节值（0~255 整数），如果未读出字节就返回 -1 。
- **int read(byte b[])** 读取 b.length 个字节到字节数组 b 中，返回实际读取的字节数。如果到达文件的末尾，则返回 -1 。
- **int read(byte b[], int off, int len)** 读取 len 个字节到字节数组 b 中，并返回实际读取的字节数目。如果到达文件的末尾，则返回 -1 ，参数 off 指定从字节数组的某个位置开始存放读取的数据。

# 10.2 文件字节输入流

例子4　使用文件字节流读取文件的内容 。

```
C:\ch10>java Example10_4
import java.io.*;
public class Example10_4 {
   public static void main(Str
      int n=-1;
      byte [] a=new byte[100];
```

图 10.4　使用文件字节流读文件

# 10.3 文件字节输出流

- 使用输出流通常包括 4 个基本步骤：

(1) 给出输出流的**目的地**

(2) 创建**指向目的地**的输出流

(3) 让输出流把数据**写入**到目的地

(4) **关闭**输出流。

# 10.3 文件字节输出流

## 1．构造方法

使用 FileOutputStream 类的下列具有刷新功能的构造方法创建指向文件的输出流。

FileOutputStream(String name);

FileOutputStream(File file);

其中：参数 name 和 file 指定的文件称为输出流的目的地

# 10.3 文件字节输出流

## 2．使用输出流写字节

输出流的 wirie 方法以字节单位向目的地写数据。

void write(int n) 向目的地写入单个字节。

void write(byte b[]) 向目的地写入一个字节数组。

void write(byte b[],int off,int len) 从字节数组中偏移量 off 处取 len 个字节写到目的地。

FileOutputStream 流顺序地写文件，只要不关闭流，每次调用 write 方法就顺序地向目的地写入内容，直到流被关闭。

# 10.3 文件字节输出流

## 3．关闭流

通过调用 close() 方法，可以保证操作系统把流缓冲区的内容写到它的目的地，即关闭输出流可以把该流所用的缓冲区的内容冲洗掉 ( 通常冲洗到磁盘文件上 ) 。

**例子5使用文件字节输出流写文件 a.txt 。**

首先使用具有刷新功能的构造方法创建指向文件 a.txt 的输出流、并向 a.txt 文件写入“新年快乐”，然后再选择使用不刷新文件的构造方法指向 a.txt，并向文件写入（即尾加）“ Happy New Year”

# 10.4 文件字符输入、输出流

1． Reader 类提供的 read 方法以字符为单位顺序地读取源中的数据。

int read()：

int read(char b[])：

int read(char b[], int off, int len)：

void close():

long skip(long numBytes):

# 10.4 文件字符输入、输出流

2． Writer 流以字符为单位顺序地写文件，每次调用 write 方法就顺序地向目的地写入内容。 Writer 类有如下常用的方法。

void write(int n): 向输出流写入一个字符。

void write(char c[]): 向输出流写入一个字符数组。

void write(char c[],int off,int length): 从给定字符数组中起始于偏移量 off 处取 len 个字符写到输出流。

void close(): 关闭输出流。

例子6使用文件字符输入、输出流将文件 a.txt 的内容尾加到文件 b.txt 中

# 10.5 缓冲流

1． BufferedReader 和 BufferedWriter 类创建的对象称作缓冲输入、输出流。二者的源和目的地必须是字符输入流和字符输出流。

- 构造方法：
  - BufferedReader(Reader in);
  - BufferedWriter (Writer out);

2.BufferedReader 和 BufferedWriter 类读写文件的方法：

- readLine() 读取文本行
- write(String s,int off,int len) 把字符串 s 写到文件中
- newLine(); 向文件写入一个回行符

# 10.5 缓冲流

由英语句子构成的文件 english.txt（每句占一行）：

The arrow missed the target.

They rejected the union demand.

Where does this road go to?

例子7 按行读取 english.txt ，并在该行的后面尾加上该英语句子中含有的单词数目，然后再将该行写入到一个名字为 englishCount.txt 的文件中。程序运行效果如图 10.5 。

```
englishCount.txt内容:
The arrow missed the target. 句子中单词个数:5
They rejected the union demand. 句子中单词个数:5
Where does this road go to? 句子中单词个数:6
```

图 10.5 使用缓冲流

# 10.6 随机流

- 使用 RandomAccessFile 类来创建一个随机访问文件流。 RandomAccessFile 类创建的流的指向既可以作为源也可以作为目的地。
- 构造方法：

  RandomAccessFile(String name,String mode) ;

  RandomAccessFile(File file,String mode) ;

# 10.6 随机流

- 相关方法：
  - seek(long a) 定位 RandomAccessFile 流的读写位置
  - getFilePointer() 获取流的当前读写位置
  - 例子8（把几个 int 型整数写入到一个名字为 tom.dat 文件 ）
  - readLine() 方法在读取含有非 ASCII 字符的文件时出现“乱码”现象的解决方法：

1．读取 String str=in.readLine();

2．用“iso-8859-1”重新编码 byte b[]=str.getBytes("iso-8859-1");

3．使用当前机器的默认编码将字节数组转化为字符串

String content=new String(b);

例子9( 使用 readLine() 读取文件 )

# 10.7 数组流

## 1．字节数组流

- 字节数组输入流 ByteArrayInputStream 和字节数组输出流 ByteArrayOutputStream 分别使用字节数组作为流的源和目标
- ByteArrayInputStream 构造方法及常用方法
  - ByteArrayInputStream(byte[] buf);
  - ByteArrayInputStream(byte[] buf,int offset,int length);
  - public int read(); 顺序地从源中读出一个字节
  - public int read(byte[] b,int off,int len); 顺序地从源中读出参数 len 指定的字节数

# 10.7 数组流

## 1．字节数组流

- ByteArrayOutputStream 流构造方法及常用方法
  - ByteArrayOutputStream();
  - ByteArrayOutputStream(int size);
  - public void write(int b); 顺序地向缓冲区写入一个字节
  - public void write(byte[] b,int off,int len); 将参数 b 中指定的 len 个字节顺序地写入缓冲区
  - public byte[] toByteArray(); 返回输出流写入到缓冲区的全部字节

# 10.7 数组流

## 2．字符数组

- CharArrayReader 和 CharArrayWriter 类是字符数组流，使用字符数组作为流的源和目标。
- 例子10 向内存（输出流的缓冲区）写入字符串 。

# 10.8 数据流

- DataInputStream 和 DataOutputStream 类创建的对象称为数据输入流和数据输出流。
- 构造方法：
  - DataInputStream （ InputStream in ）创建的数据输入流指向一个由参数 in 指定的底层输入流
  - DataOutputStream （ OutnputStream out ）创建的数据输出流指向一个由参数 out 指定的底层输出流

例子11　写几个 Java 类型的数据到一个文件。

例子12　将字符串加密后写入文件，如图 10.6 所示

```
C:\ch10>java Example10_12
加密命令:建涔愎斟晨陈暘?杭江?哄暨??烯
解密命令:度江总攻时间是4月22日晚10点
```

图 10.6　使用数据流加密信息

# 10.9 对象流

- ObjectInputStream 和 ObjectOutputStream 类创建的对象称为对象输入流和对象输出流。
- 它的构造方法是：
  - ObjectInputStream(InputStream in)
  - ObjectOutputStream(OutputStream out)
  - 相关方法：
  - writeObject(Object obj) 将一个对象 obj 写入到一个文件
  - readObject() 读取一个对象到程序中
- 所谓序列化：一个类如果实现了 Serializable 接口，那么这个类创建的对象就是所谓序列化的对象。

# 10.9 对象流

例子13

Example10_13.java , TV.java，使用对象流读写 TV 类创建的对象，效果如图 10.7 所示。

```
C:\ch10>java Example10_13
changhong的名字:长虹电视
changhong的价格:5678
xinfei的名字:新飞电视
xinfei的价格:6666
```

图 10.7 使用对象流读写对象

# 10.10 序列化与对象克隆

- 如果一个“复制品”实体的变化不会引起原对象实体发生变化，反之亦然。这样的复制品称为原对象的一个克隆对象或简称克隆。
- 一个对象调用 clone() 方法就可以获取该对象的克隆对象。
- 对象输入流通过对象的序列化信息来得到当前对象的一个克隆。
- 例子14（单击“写出对象”按扭将标签写入到内存，单击“读入对象”按扭读入标签的克隆对象 ）

# 10.11 使用 Scanner 解析文件

**使用 Scanner 类和正则表达式来解析文件。**

**1．使用默认分隔标记解析文件**

**创建 Scanner 对象，并指向要解析的文件，例如：**

```
File file = new File("hello.java");
Scanner sc = new Scanner(file);
```

**sc 将空白作为分隔标记**

**相关方法** **next()** 依次返回 file 中的单词

**hasNext()** 判断 file 最后一个单词是否已被 next() 方法返回 .

**例题15** **解析文件 cost.txt中的全部消费**

# 10.11 使用 Scanner 解析文件

**2．使用正则表达式作为分隔标记解析文件：** 创建 Scanner 对象，指向要解析的文件，并使用 useDelimiter 方法指定正则表达式作为分隔标记，例如：

```
File file = new File("hello.java");
Scanner sc = new Scanner(file);
sc.useDelimiter( 正则表达式 );
```

sc 将正则表达式作为分隔标记

相关方法 next() 依次返回 file 中的单词

hasNext() 判断 file 最后一个单词是否已被 next() 方法返回

例子16 解析 student.txt文件中的学生成绩

# 10.12 文件对话框

- 构造方法 JFileChooser() 创建初始不可见的有模式的文件对话框。然后文件对话框调用下述 2 个方法：
  - showSaveDialog(Component a);
  - showOpenDialog(Component a);
- 都可以使得对话框可见，只是呈现的外观有所不同， showSaveDialog 方法提供保存文件的界面， showOpenDialog 方法提供打开文件的界面。上述两个方法中的参数 a 指定对话框可见时的位置，当 a 是 null 时，文件对话框出现在屏幕的中央；如果组件 a 不空，文件对话框在组件 a 的正前面居中显示。

例子 17　Example10_17java，WindowReader.java ，使用文件对话框打开和保存文件

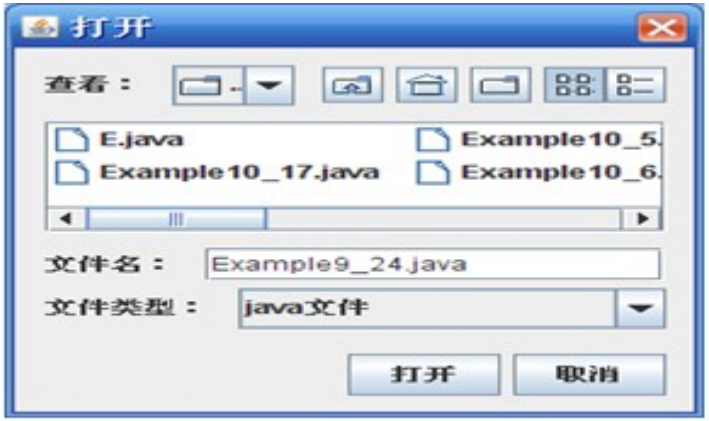


图 10.10　文件对话框

# 10.13 带进度条的输入流

- 如果读取文件时希望看见文件的读取进度可以使用 javax.swing 包提供的输入流类： ProgressMonitorInputStream 。
- 它的构造方法：
  - ProgressMonitorInputStream(Conmponent c,String s, InputStream);
- 例子18

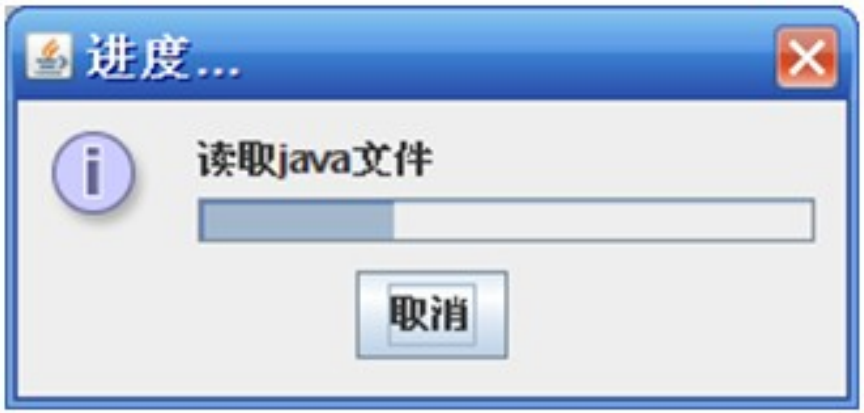


图 10.12 进度条

# 10.14 文件锁

- FileLock 、 FileChannel 类处理 Java 提供的文件锁功能。它们分别在 java.nio 和 java.nio.channels 包中。
- 输入、输出流读写文件时可以使用文件锁。

# 10.14 文件锁

- RondomAccessFile 创建的流在读写文件时使用文件锁的步骤如下：

  1．先使用 RondomAccessFile 流建立指向文件的流对象，该对象的读写属性必须是 rw，例如：

  ```
  RandomAccessFile input=new
        RandomAccessFile("Example.java","rw");
  ```

  2． input 流调用方法 getChannel() 获得一个连接到地层文件的 FileChannel 对象（信道），例如

  ```
  FileChannel channel=input.getChannel();
  ```

  3．信道调用 tryLock() 或 lock() 方法获得一个 FileLock （文件锁）对象，这一过程也称作对文件加锁 .

  例如： FileLock lock=channel.tryLock();

- 例子19 (Example10_19.java , WindowFileLock.java )

# 10.15 应用举例

## 1. 标准化考试

标准化试题文件的格式要求如下：
每道题目提供 A 、 B 、 C 、 D 四个选择（单项选择）。
两道题目之间是用减号（ - ）尾加前一题目的答案分隔 ( 例如： ----D-----)

1. 北京奥运是什么时间开幕的？
   A. 2008-08-08 B. 2008-08-01
   C. 2008-10-01 D. 2008-07-08
------A------
2. 下列哪个国家不属于亚洲？
   A. 沙特 B. 印度 C. 巴西 D. 越南
------C------
3. 2010 年世界杯是在哪个国家举行的？
   A. 美国 B. 英国 C. 南非 D. 巴西
------C-----

**例子 20** Example10_20.java , StandardExam.java 每次读取试题文件中的一道题目，并等待用户回答，用户做完全部题目后，程序给出用户的得分

```
1. 北京奥运是什么时间开幕的?
   A.2008-08-08  B. 2008-08-01
   C.2008-10-01  D. 2008-07-08

输入选择的答案:A
2. 下列哪个国家不属于亚洲?
   A.沙特  B.印度  C.巴西  D.越南

输入选择的答案:C
3. 2010年世界杯是在哪个国家举行的?
   A.美国  B.英国  C.南非  D.巴西

输入选择的答案:C
4. 下列哪种动物属于猫科动物?
   A.鬣狗  B.犀牛  C.大象  D.狮子

输入选择的答案:D
最后的得分:4
```

图 10.12 标准化考试

# 10.15 应用举例

## 2. 通讯录

下面的例子 21(Example10_21.java，InputArea.java，CommFrame.java ) 使用 RandomAccessFile 流实现一个通讯薄的录入与显示系统，其中的 InputArea.java 源文件中的类负责通讯薄信息的录入，CommFram 窗体组合了 InputArea 类的实例。通讯录效果如图 10.13 、 10.14 。


图 10.13 录入界面


图 10.14 显示界面

# 10.15 应用举例

## 3. 简单的 Java 集成开发环境（ IDE ）

例 子 22(Example10_22.java , JDKWindow.java , CompileDialog.java , RunDialog.java ) 是一个用于编译和运行 java 程序的小软件（代替在命令行窗口运行 javac.exe 和 java.exe ） , 即使用 GUI 程序来编译、运行 Java 应用程序 .

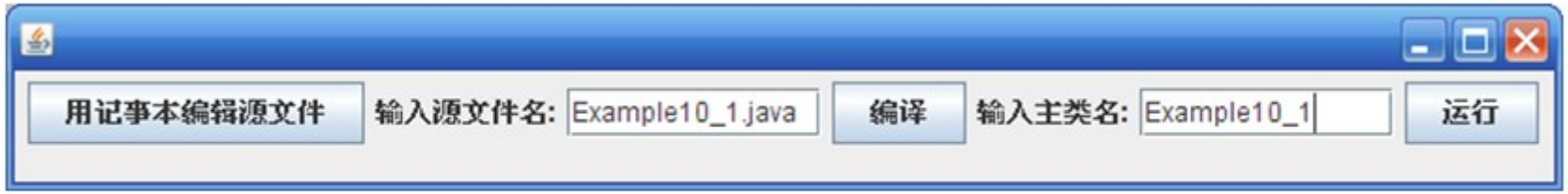


图 10.15 输入源文件名或主类名