Java的流及文件处理

Java中提供了两种方式作为输入/输出的API：流及文件。

读写数据又可以通过两种格式来进行：二进制格式和文本格式。

# 流

输入流：可以从其中读入一个字节序列的对象；

输出流：可以向其中写入一个字节序列的对象；

## 流家族

按照使用方法进行划分，形成了处理字节和字符的两个单独的层次结构。

### 处理字节

InputStream和OutputStream类可以读写单个字节或字节数组。子类用来读写字符串和数字。例如，DataInputStream和DataOutputStream可以以二进制格式读写所有的基本Java类型；ZipInputStream和ZipOutputStream可以以常见的Zip压缩格式读写文件。



#### InputStream接口

abstract int read();

从数据中读入一个字节，并返回该字节。碰到流结尾时返回-1。

int read(byte[] b);

int read(byte[] b, int off, int len);

读入一个字节数组，并返回实际读入的字节数，碰到流结尾时返回-1。

off：第一个读入字节应该被放置的位置在b中的偏移量；

len：读入字节的最大数量；

long skip(long n);

在输入流中跳过n个字节，返回实际跳过的字节数(如果碰到流的结尾，则可能小于n)；

int available();

返回在不阻塞的情况下课获取的字节数；

void close();

#### OutputStream接口

abstract void write(int n);

写出一个字节的数据；

void write(byte[] b);

void write(byte[] b, int off, int len);

写出数组b中所有字节或者某个范围的字节；

off：第一个写出字节在b中的偏移量；

len：写出字节的最大数量；

void close();

void flush();

### 处理字符

对于Unicode文本，使用Reader和Writer的子类进行读写。从其继承的类拥有的读入和写出操作都是基于两字节的Unicode码元的，而不是基于单字节的字符。



### 附加接口



#### Closeable

拥有方法：void close() throws IOException;

Closeable接口扩展了java.lang.AutoCloseable接口，因此，对任何Closeable进行操作时，都可以使用try-with-resource语句。

#### Flushable

拥有方法：void flush();

#### Readable

拥有方法：int read(CharBuffer cb);

尝试着向cb读入其可持有数量的char值。返回读入的char值的数量，或者当从这个Readable中无法再获得更多的值时返回-1。

CharBuffer类拥有按顺序和随机地进行读写访问的方法，它表示一个内存中的缓冲区或者一个内存映象的文件。

#### Appendable

拥有两个用于添加单个字符和字符序列的方法：

Appendable append(char c);

Appendable append(CharSequence s);

向这个Appenable中追加给定的码元或者给定的序列中的所有码元，返回this。

CharSequence接口描述了一个char值序列的基本属性。String、CharBuffer、StringBuilder和StringBuffer都实现了它。

### 组合流过滤器

某些流（例如FileInputStream和由URL类的openStream方法返回的输入流）可以从文件和其他更外部的位置上获取字节，而其他的流（例如DataInputStream和PrintWriter）可以将字节组装到更有用的数据类型中。

Java中通过嵌套过滤器来添加多重功能。FilterInputStream和FilterOutputStream的子类用于向原生字节流添加额外的功能。例如：

DataInputStream din = new DataInputStream(

new BufferedInputStream(

new FileInputStream(“employee.dat”)

)

);

## 文本输入输出

存储文本字符串时，需要考虑字符编码的方式。

OutputStreamWriter类将使用选定的字符编码方式，把Unicode字符流转换为字节流。而InputStreamReader类将包含字节（用某种字符编码方式表示的字符）的输入流转换为可以产生Unicode码元的读入器。可以在类的构造器中进行指定的方式来选择不同的编码方式。

### 输出

使用PrintWriter进行文本输出。这个类拥有以文本格式打印字符串和数字的方法。还有一个将PrintWriter链接到FileWriter的便捷方法。

PrintWriter out = new PrintWriter(“employee.bat”);

等同于

PrintWriter out = new PrintWriter(new FileWriter(“employee.bat”));

使用print、println和printf方法打印数字、字符、boolean值、字符串和对象。

### 输入

使用Scanner来读入文本输入。也可以使用BufferedReader类处理文本输入，但BufferedReader没有任何用于读入数字的方法，所以推荐使用Scanner。

### 字符集

## 二进制输入输出

# 文件