Java输入输出流

本章知识点

- Java流的类层次结构
- File类和RandomAccessFile类
- 字节流
  - 抽象类InputStream和OutputStream
  - 文件流FileInputStream和FileOutputStream
  - 缓冲流BufferedInputStream和BufferedOutputStream
  - 数据过滤流DataInputStream和DataOutputStream
  - 打印流PrintStream
  - 序列化接口Serializable
  - 对象流ObjectInputStream和ObjectOutputStream
  - 字节数组流ByteArrayInputStream和ByteArrayOutputStream

本章知识点

- 字符流
  - 抽象类Reader和Writer
  - 转换流InputStreamReader和OutputStreamWriter
  - FileReader和FileWriter
  - BufferedReader类
  - PrintWriter类

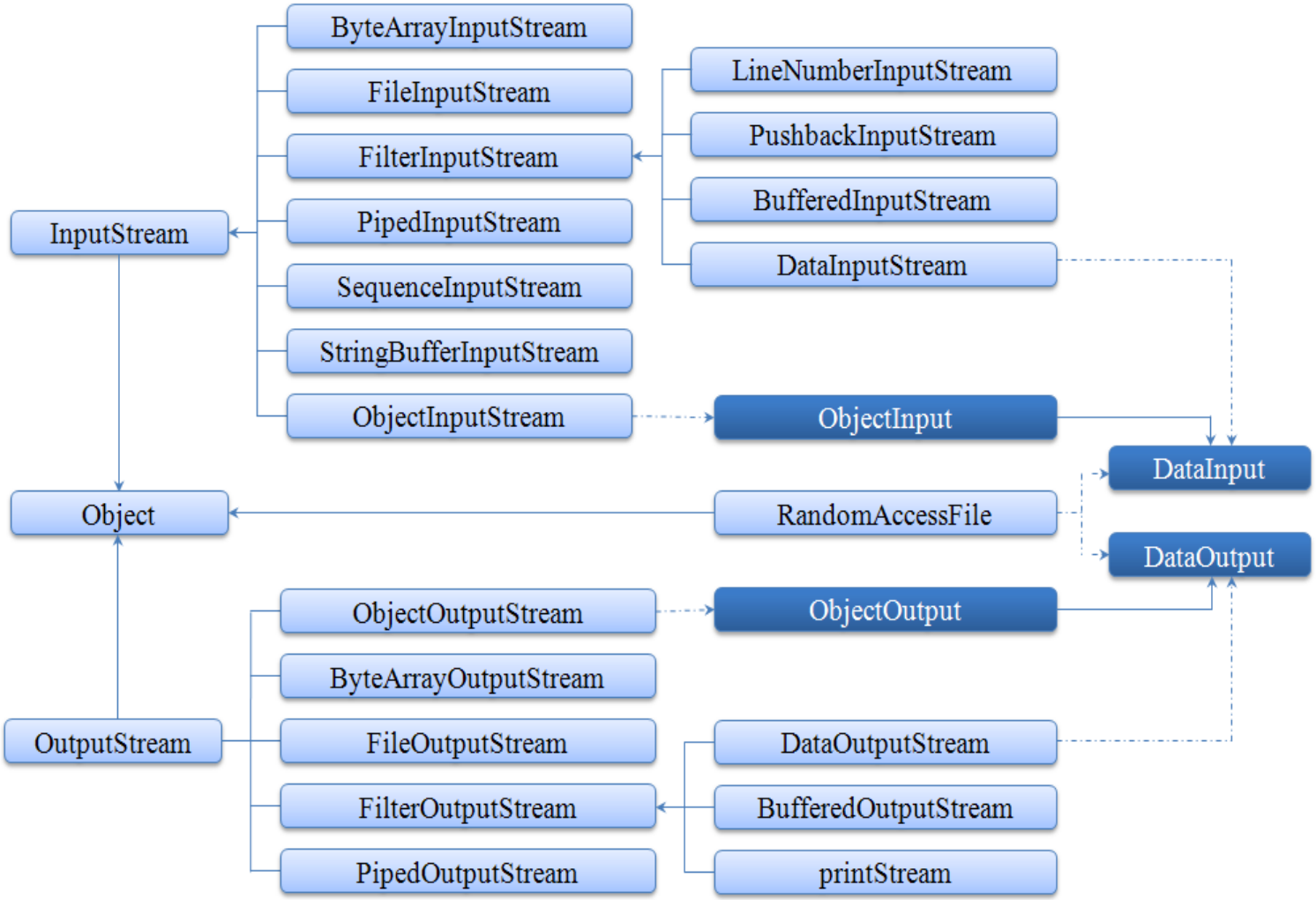
输入输出流

- Java中的输入/输出（Input/Output，简称I/O）：程序与外部设备或其他计算机进行数据交换的过程。

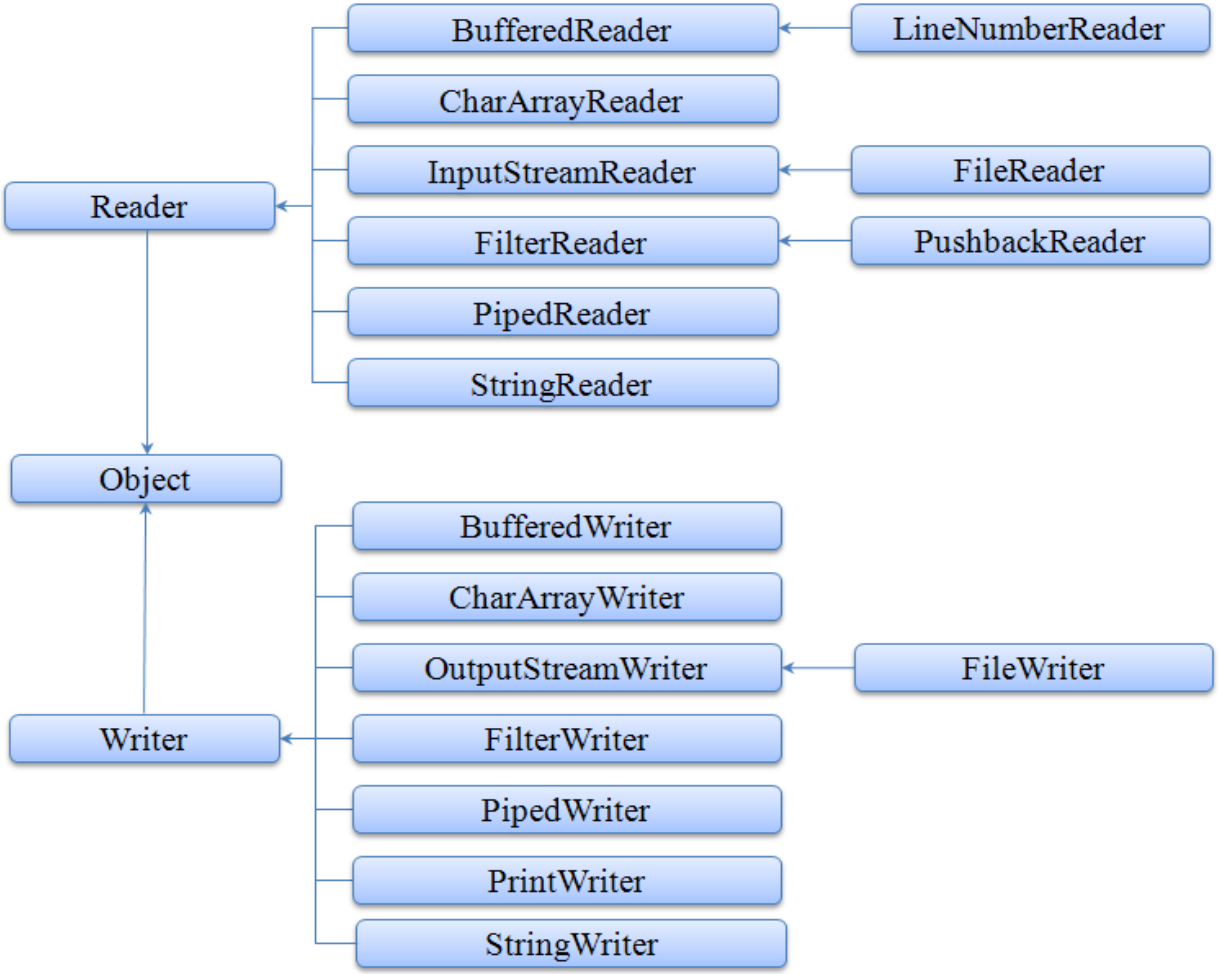
- Java用“流”的方式处理I/O，对应不同类型的I/O问题，会有相应的流对象提供解决方案。
  - JDK 1.0版本中设计了面向字节的I/O流。
  - JDK 1.1 对基本的I/O流类库进行了重大的修改，增加了面向字符的I/O流，这些流对应的类和接口位于java.io包下。
  - 在JDK1.4中添加了nio类用于改进性能和功能，这些类位于java.nio包下。

12.1 Java流的类层次结构

字节流层次结构


ByteArrayInputStream
FileInputStream
FilterInputStream
PipedInputStream
SequenceInputStream
StringBufferInputStream
ObjectInputStream
LineNumberInputStream
PushbackInputStream
BufferedInputStream
DataInputStream
InputStream
ObjectInput
DataInput
Object
RandomAccessFile
DataOutput
ObjectOutputStream
ObjectOutput
ByteArrayOutputStream
OutputStream
FileOutputStream
DataOutputStream
FilterOutputStream
BufferedOutputStream
PipedOutputStream
printStream

字符流层次结构


Reader
BufferedReader
LineNumberReader
CharArrayReader
InputStreamReader
FileReader
FilterReader
PushbackReader
PipedReader
StringReader
Object
Writer
BufferedWriter
CharArrayWriter
OutputStreamWriter
FileWriter
FilterWriter
PipedWriter
PrintWriter
StringWriter

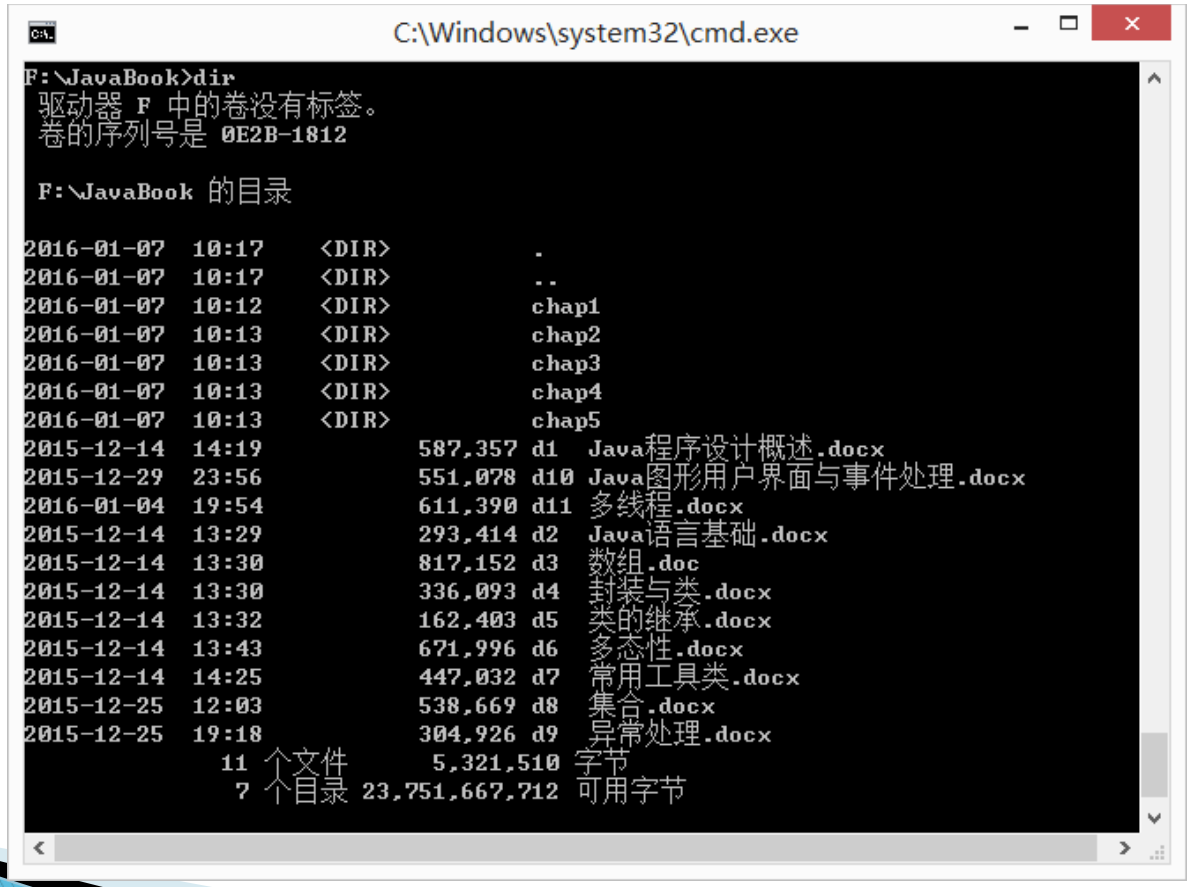
12.2 文件

- File：用于管理文件系统中有关文件/文件夹的信息
- RandomAccessFile：用区别于输入/输出流的行为方式访问文件

12.2.1 File类

File类：用于表示文件系统中的文件/文件夹

【例12-1】实现Windows环境下的dir命令。



12.2.2 RandomAccessFile类

1．RandomAccessFile的特点

- RandomAccessFile类以字节为单位进行文件内容的存取。

- （1）RandomAccessFile提供了文件的“随机访问”方式，在RandomAccessFile类中定义了文件记录指针，标识当前正在读写的位置，它可以指向文件中的任意位置。
- （2）RandomAccessFile既可以读取文件的内容，也可以向文件输出数据，集读、写功能于一身。

12.2.2 RandomAccessFile类

2．RandomAccessFile中的读写方法

- 查看API文档

3、RandomAccessFile中的其他方法

- long getFilePointer()：返回文件记录指针的当前位置。
- void seek(long pos)：将文件记录指针定位在pos位置。

- 通过它们可以获取和操作文件记录指针，实现RandomAccessFile与众不同的定位随机访问。

12.2.2 RandomAccessFile类

4．RandomAccessFile的常用访问方式

- 创建RandomAccessFile对象时，需要指定文件的访问方式，常用的包括下面两种：
- “r”：以只读方式打开指定文件。
- “rw”：以读、写方式打开指定文件，如果文件不存在则尝试创建该文件。

- RandomAccessFile没有只写的访问方式。

12.2.2 RandomAccessFile类

【例12-2】利用RandomAccessFile对象向data.dat文件中写入字符和int型数据，并检测每次写入后文件记录指针的取值。

【例12-3】向磁盘“demo”文件夹的data.dat文件中写入10个int型随机整数；从键盘上输入1~10这些序号，在控制台打印出对应数字。

【例12-4】利用RandomAccessFile实现文件的多线程下载。

12.3 字节流

- 字节流以8位的字节为读写单位。
- 字节流体系：以抽象类InputStream和OutputStream作为顶层向下包含了众多子类成员，可以按照输入流/输出流的方式完成各种基本数据类型数据、数组、字符串、对象、文件等的读入和写出。

12.3.1 抽象类InputStream和OutputStream

1、InputStream类

- 用来表示那些从不同数据源产生输入的类。
- InputStream的主要子类
  - ByteArrayInputStream：把内存中的一个缓冲区作为InputStream，读取该缓冲区内容。
  - StringBufferInputStream：把一个字符串对象转换成InputStream。
  - FileInputStream：把文件作为InputStream，用于从文件中读取信息。
  - PipedInputStream：实现了管道（pipe）的概念，产生用于写入PipedOutputStream的数据，在线程通信中使用。
  - SequenceInputStream：把多个InputStream合并为一个InputStream。

12.3.1 抽象类InputStream和OutputStream

- InputStream中的方法
  - int read()
  - int read(byte[] b)
  - int read(byte[] b, int off, int len)
  - int available()：返回流中可用字节数。
  - void close()：关闭输入流，并释放与该流关联的所有系统资源。

12.3.1 抽象类InputStream和OutputStream

2. OutputStream

- 用于表示输出要去往的目标
- OutputStream主要的子类
  - ByteArrayOutputStream：在内存中创建缓冲区，所有送往“流”的数据都先放置在此缓冲区。
  - FileOutputStream：用于将信息写入文件。
  - PipedOutputStream：配合PipedInputStream类在线程间通信。

12.3.1 抽象类InputStream和OutputStream

- OutputStream类中的方法
  - write(int b)
  - write(byte[] b)
  - write(byte[] b, int off, int len)
  - void close()：关闭输出流。
  - void flush()：强制刷新输出缓冲区内容，将输出缓冲区内容写入流。

12.3.2 文件流FileInputStream和FileOutputStream

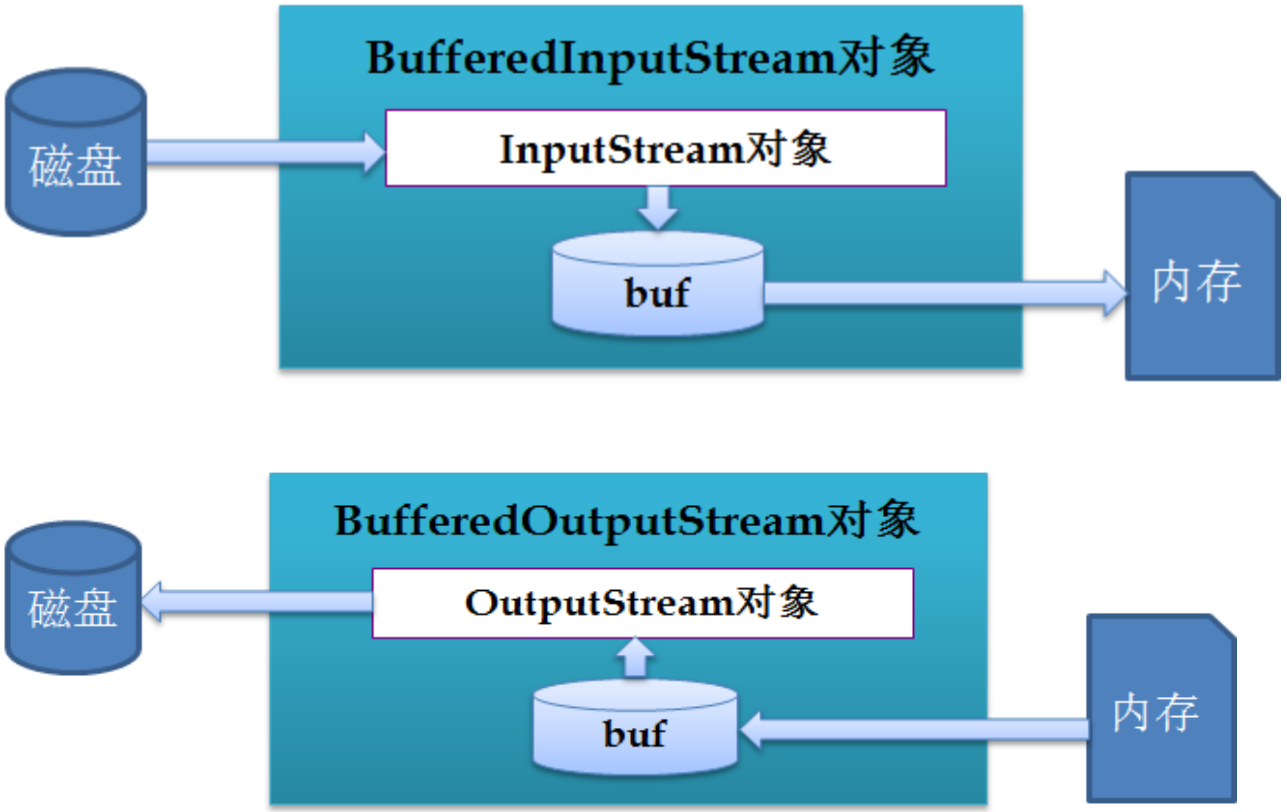
- FileInputStream和FileOutputStream：基于字节、广泛用于操作文件。
  - FileInputStream类用于读取一个文件，FileOutputStream类用于将数据写入一个文件。
  - FileInputStream常用构造方法
    - FileInputStream(String name)
    - FileInputStream(File file)

- 它们都会打开一个到实际文件的连接，并创建一个文件输入流，如果该文件不存在，或者它是一个目录，或者因为其他原因而无法打开，会抛出FileNotFoundException异常。前者文件通过路径名name指定，后者通过File对象指定。

12.3.3 缓冲流BufferedInputStream和BufferedOutputStream

- BufferedInputStream的构造方法：
  - BufferedInputStream(InputStream in)：包装InputStream对象、创建BufferedInputStream对象，并创建一个默认大小（8192字节）的字节数组做缓冲区。
  - BufferedInputStream(InputStream in, int size)：包装InputStream对象、创建指定缓冲区大小的BufferedInputStream对象。

12.3.3 缓冲流BufferedInputStream和BufferedOutputStream



12.3.3 缓冲流BufferedInputStream和BufferedOutputStream

- 【例12-6】对比带缓冲区的文件复制与不带缓冲区文件复制的性能。
  - 通过完成相同的复制工作所花费的时间对比得到带缓冲区与不带缓冲区间的差别。

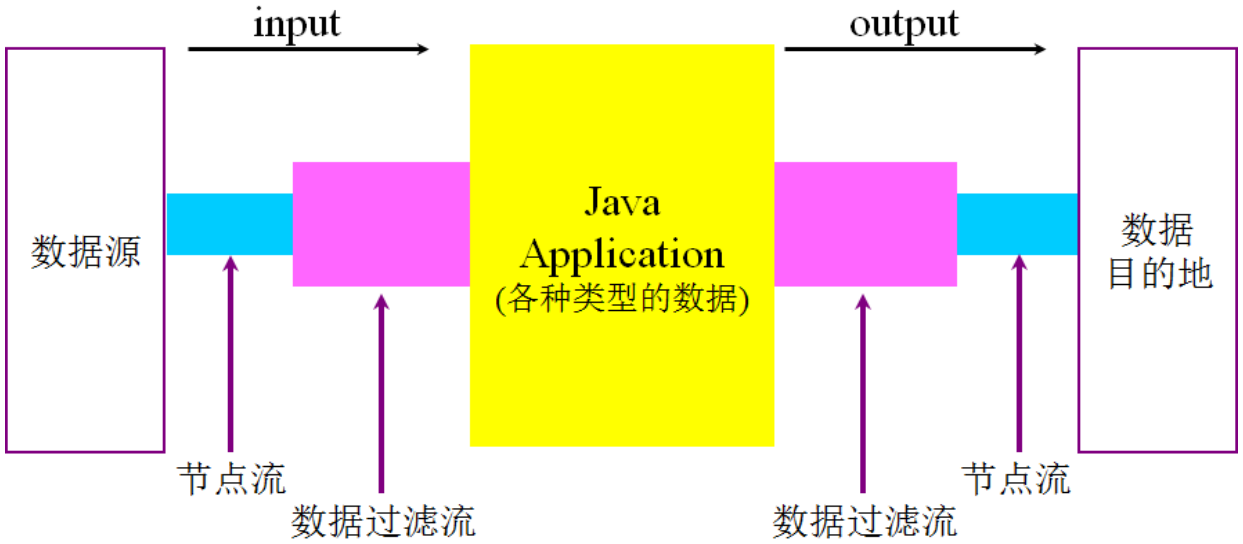
12.3.4 数据过滤流DataInputStream和DataOutputStream

- 数据过滤流类DataInputStream和DataOutputStream是FilterInputStream和FilterOutputStream的子类。

- 它们将字节数据按照指定形式结合成有意义的基本数据类型数据、以及String类型数据。

12.3.4 数据过滤流DataInputStream和DataOutputStream

- 它们的构造方法就是对已有的InputStream和OutputStream对象进行包装：DataInputStream(InputStream in), DataOutputStream(OutputStream out)



12.3.4 数据过滤流DataInputStream和DataOutputStream

【例12-7】使用数据过滤流将1000-2000间的素数写到文件中持久化保存；将这些素数从文件读出、按每行10个的形式打印。

- 分析：使用自定义方法boolean isPrime(int)判断某数是否是素数。找到素数后，将其用writeInt()方法写入文件。读取文件时，每次使用readInt()方法读取一个int。

12.3.5 打印流PrintStream

- PrintStream提供了大量重载的print()和println()方法，可以打印各种数据值。
  - print (boolean)
  - print(char)
  - print (char[])
  - print (double)
  - print (float)
  - print (long)
  - print (int)
  - print (Object)
  - print(String)

  - println()方法在print()的基础上，每次输出完毕后打印一个回车换行。

12.3.6 序列化接口Serializable与对象流ObjectInputStream和ObjectOutputStream

- 有时希望将以对象方式存在于内存中的数据存储至文件（持久化到文件），需要时再将其从文件中读出还原为对象，或者在网络上传送对象，这时可以使用Java提供的对象流ObjectInputStream和ObjectOutputStream。

12.3.6 序列化接口Serializable与对象流ObjectInputStream和ObjectOutputStream

【例12-8】将一个Student对象持久化到文件，并从文件读出、打印。

12.3.7 字节数组流ByteArrayInputStream和ByteArrayOutputStream

- 构造方法：
  - ByteArrayInputStream(byte[] buf)：使用buf作为其缓冲区数组创建一个 ByteArrayInputStream对象。
  - ByteArrayInputStream(byte[] buf, int offset, int length)：使用buf数组从offset位置开始的length长度空间作为缓冲区创建 ByteArrayInputStream。
  - ByteArrayOutputStream()：创建一个新的字节数组输出流。
  - ByteArrayOutputStream(int size)：创建一个新的字节数组输出流，它具有指定大小的缓冲区容量。

12.3.7 字节数组流ByteArrayInputStream和ByteArrayOutputStream

【例12-9】实现对象的深复制。

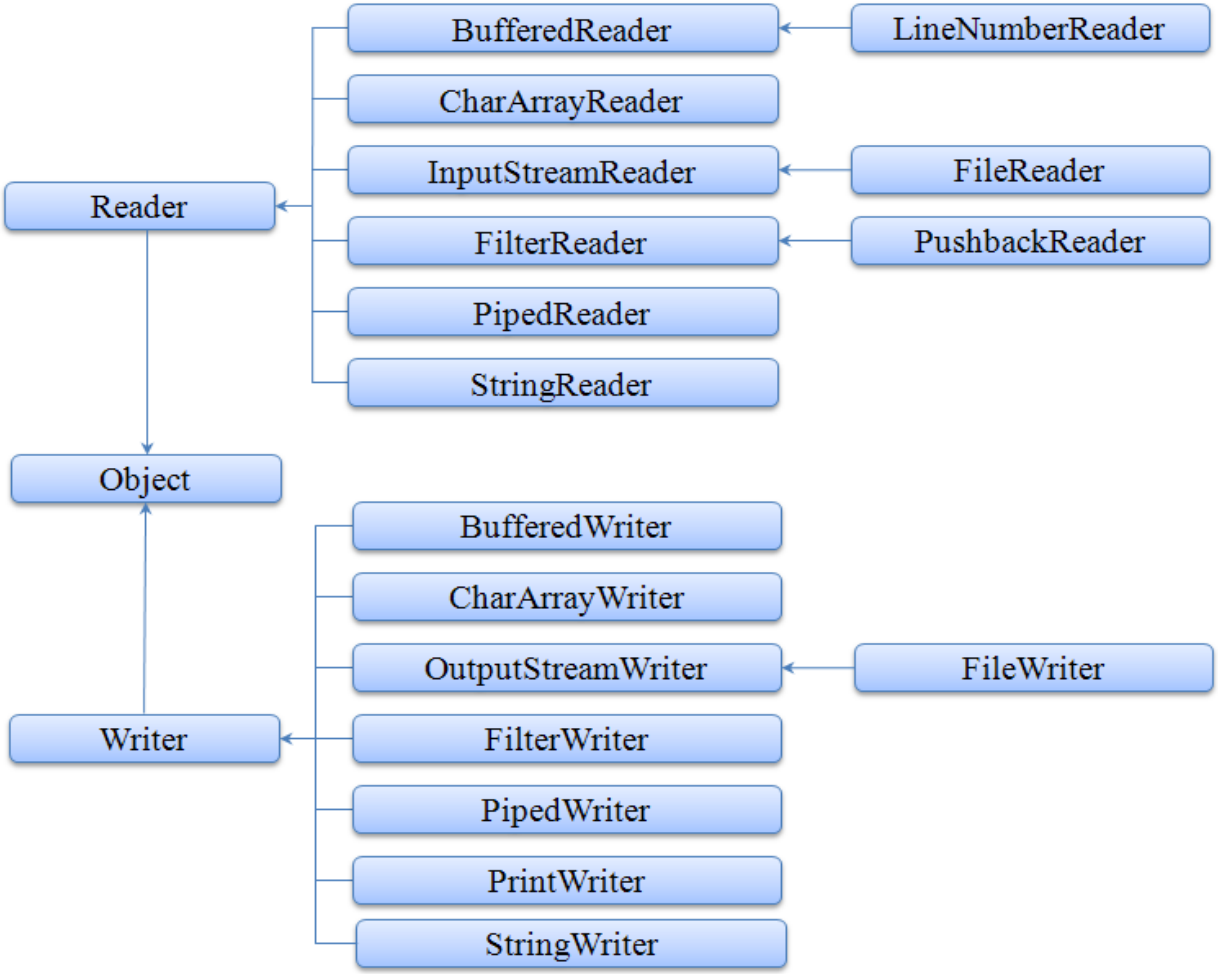
- 深复制的前提：对象以及对象内部所有引用到的对象都是可序列化的，即都实现了Serializable接口。
- 将对象持久化到文件中再读出可以获得原对象的一个拷贝，所以利用这个方法可以深复制对象。

12.4 字符流

- InputStream和OutputStream系列处理的是字节流，存取数据流时以字节为单位，但它们在读写文本字符、字符串时就不太方便。另外，字节流只支持ISO-8859-1编码，而Java本身使用Unicode编码，各个平台、系统中的字符还会使用其他编码方式。

- 为了便于读写字符型数据，让Java的I/O流都支持Unicode编码，并允许使用其他字符编码方案，Java 1.1的类库中增加了Reader和Writer继承层次结构，分别表示字符输入流和字符输出流。

12.4.1 抽象类Reader和Writer



12.4.2 转换流InputStreamReader和OutputStreamWriter

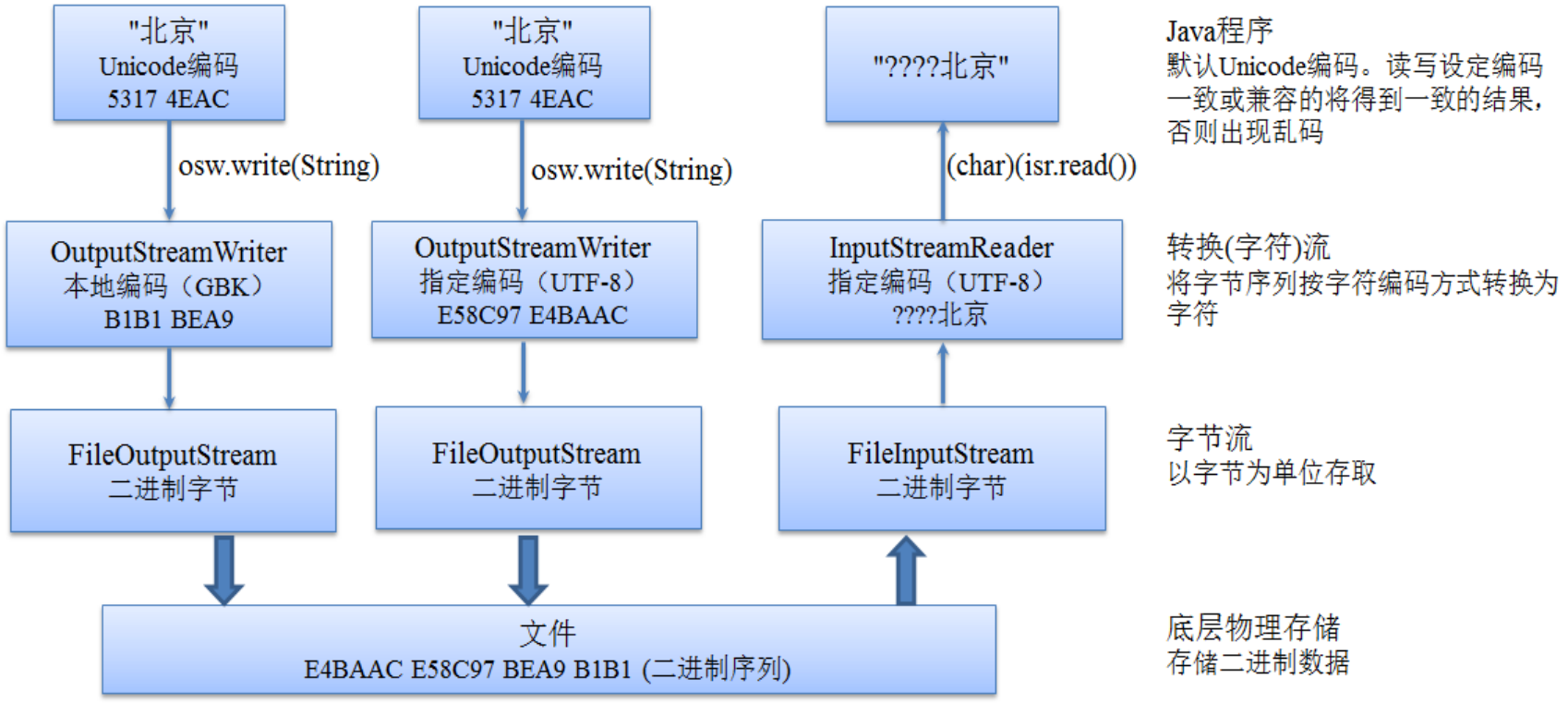
- **1．关于字符编码**
- （1）ASCII
- （2）汉字编码
- （3）国际编码
- （4）ISO-8859-1（ISO拉丁字母表，也称ISO-LATIN-1）

12.4.2 转换流InputStreamReader和OutputStreamWriter

- 2．利用转换流设置字符编码
  - 如果需要输入、输出流采用特定编码方案，可以使用InputStreamReader和OutputStreamWriter类，它们在将字节流转换为字符流的同时，可以指定字符编码方式。

12.4.2 转换流InputStreamReader和OutputStreamWriter

【例12-10】向文件中写入一个中文字符串，再将其从文件读出。



12.4.3 FileReader和FileWriter

- 如果存取的是一个文本文件，可以直接使用FileReader和FileWriter类，它们分别继承自InputStreamReader和OutputStreamWriter。

- 【例12-11】将九九乘法表保存在文本文件中。

12.4.4 BufferedReader类

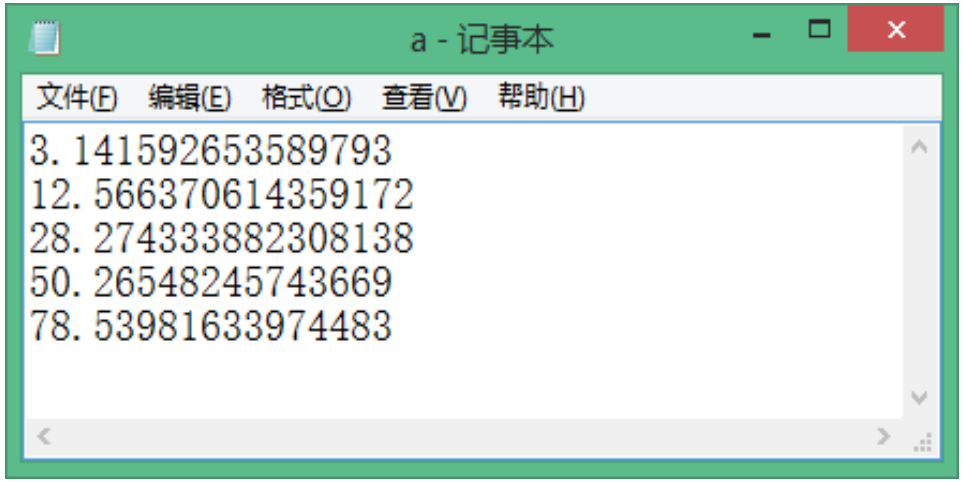
- BufferedReader和BufferedWriter类都带有8192个字符的缓冲区，缓冲区的作用与BufferedInputStream和BufferedOutputStream类相同。
  - BufferedReader类还可以“文本行”为基本单位读取数据，文本行是以回车换行结束的字符序列。
  - String readLine()：从输入流中读取一行字符，如果读遇到流结束，则返回null。

12.4.5 PrintWriter类

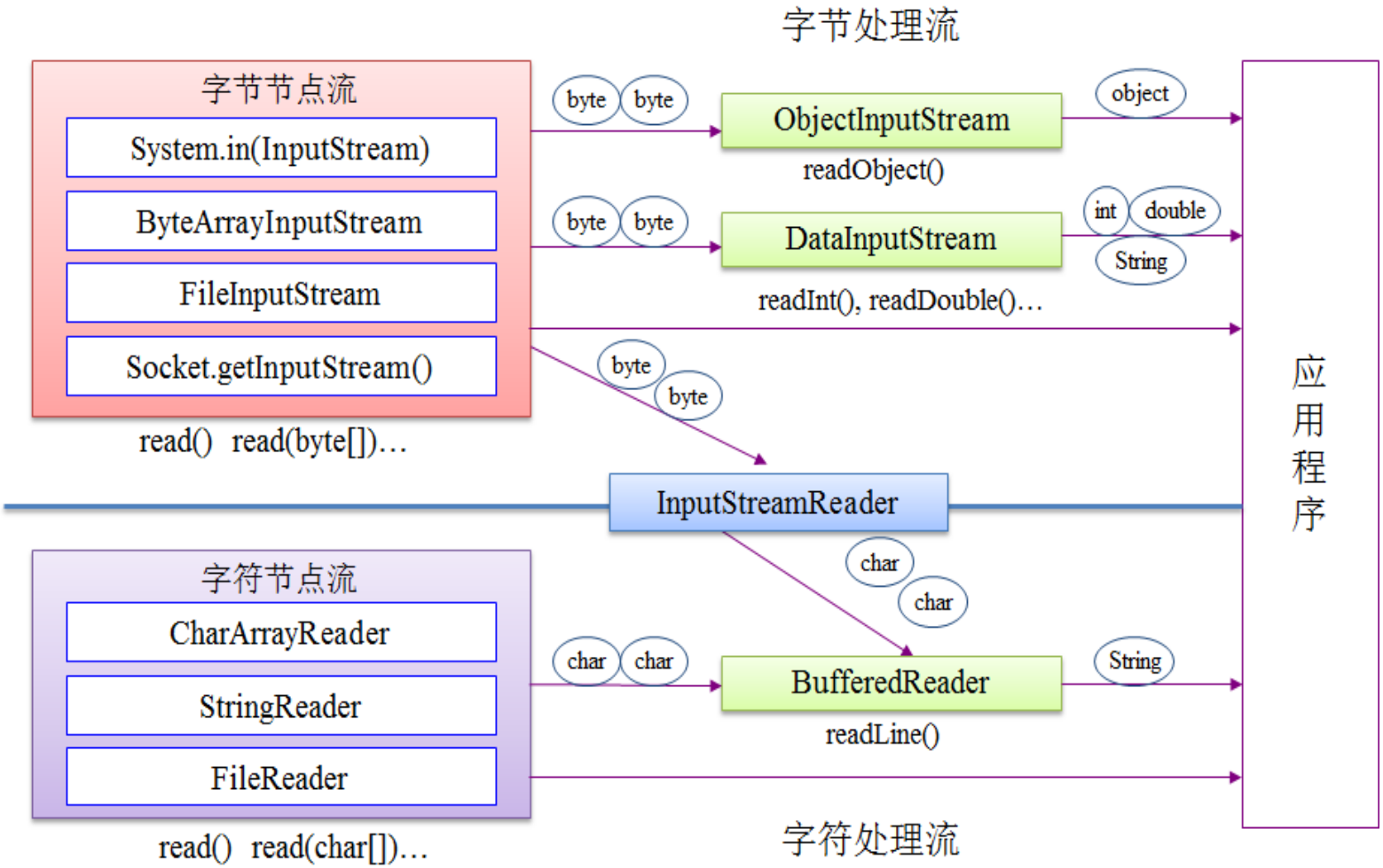
- PrintStream的问题是它不支持国际化，不能用与平台无关的方式处理换行。所以在JDK 1.1中引入了PrintWriter类，它是字符流，依旧使用与PrintStream相同的格式化接口print()和println()方法，但提供了国际化支持。在输出方面，PrintWriter比PrintStream更为合适。

12.4.5 PrintWriter类

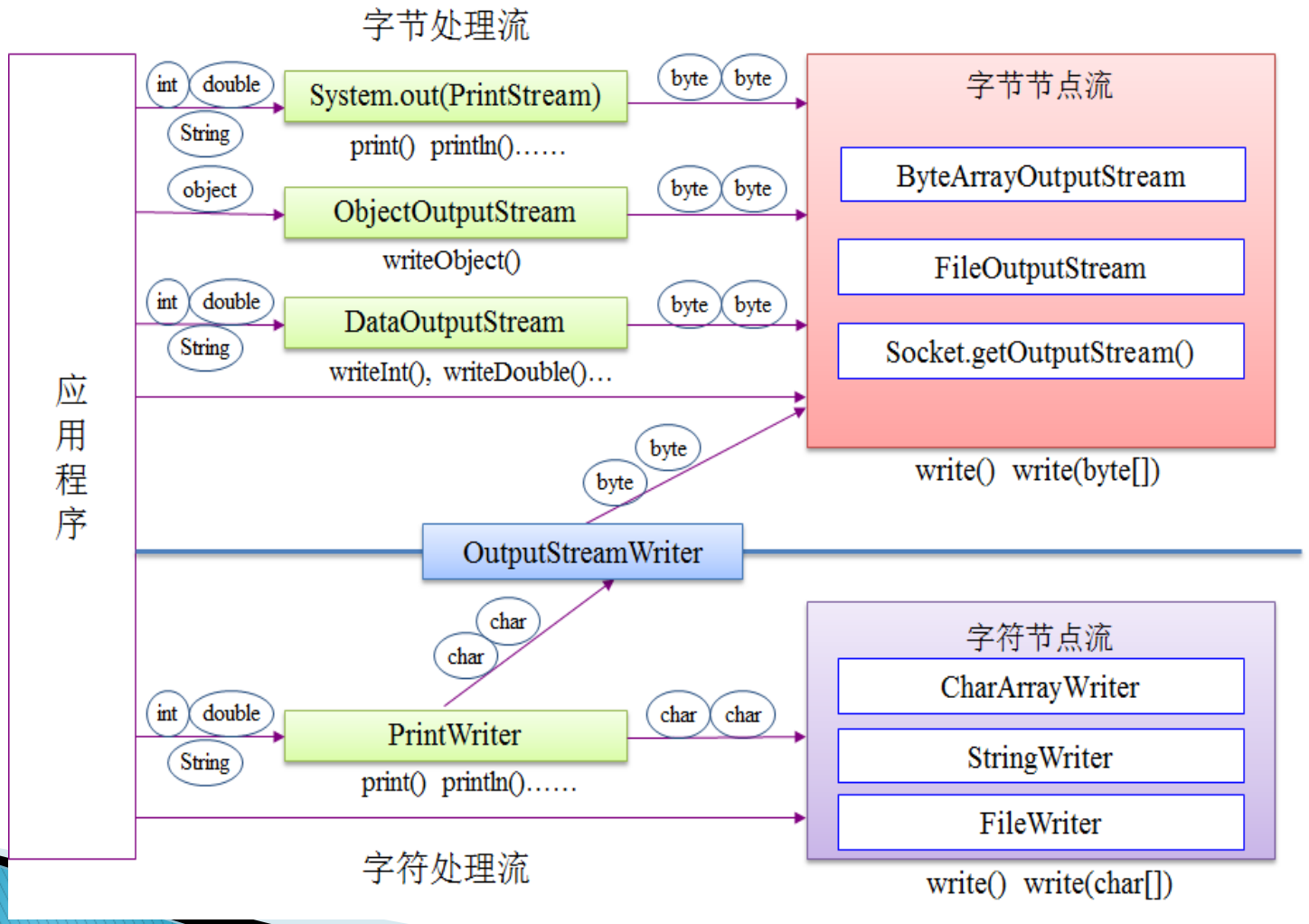
【例12-12】将计算得到的几个圆的面积写入一个文件；再将文件中的信息读取出来在控制台打印。



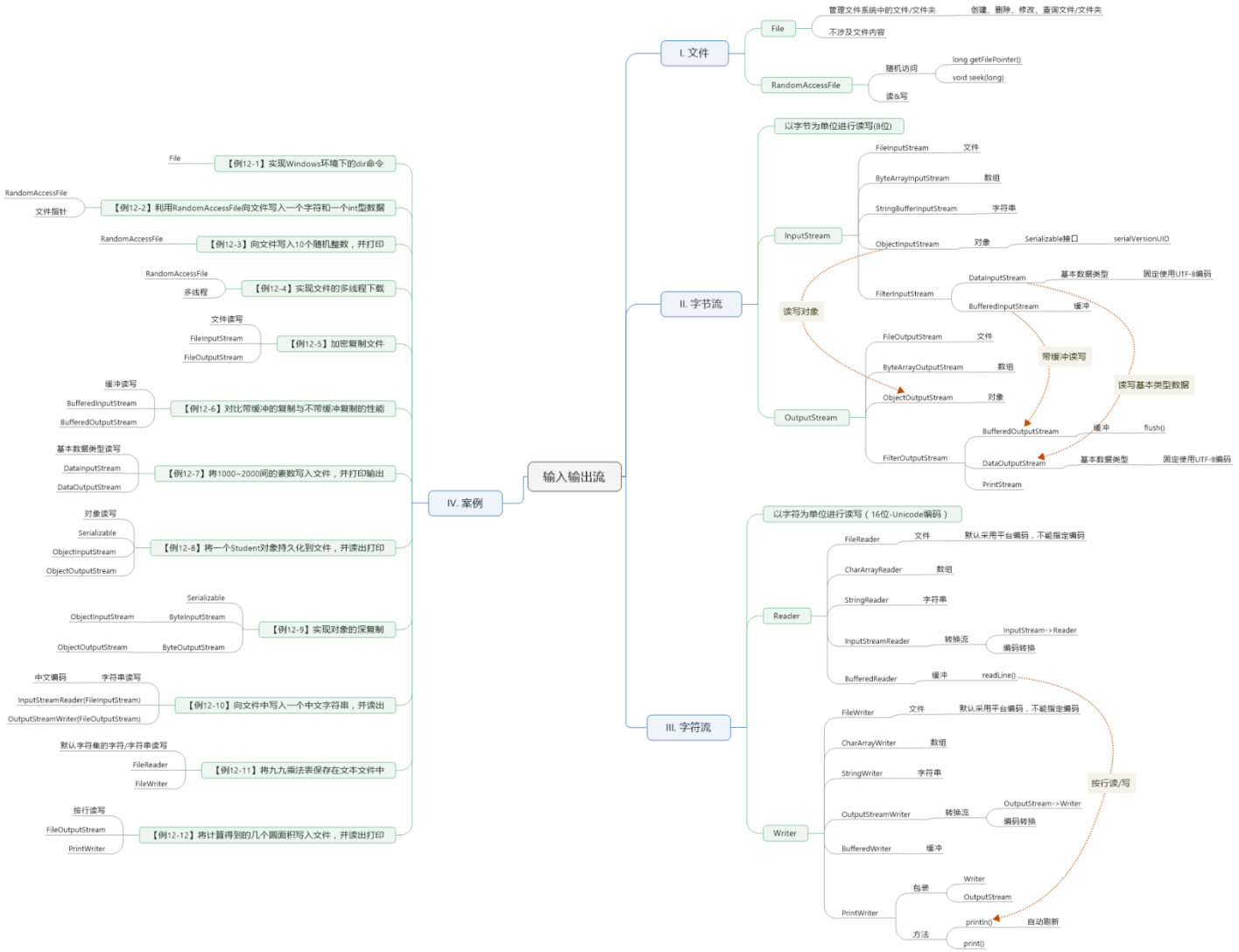
12.5 Java输入流输出流汇总



12.5 Java输入流输出流汇总



本章思维导图


输入输出流
I. 文件
File
管理文件系统中的文件/文件夹
创建、删除、修改、查询文件/文件夹
不涉及文件内容
RandomAccessFile
随机访问
long getFilePointer()
void seek(long)
读&写
II. 字节流
以字节为单位进行读写(8位)
InputStream
FileInputStream 文件
ByteArrayInputStream 数组
StringBufferInputStream 字符串
ObjectInputStream 对象 Serializable接口 serialVersionUID
FilterInputStream
DataInputStream 基本数据类型 固定使用UTF-8编码
BufferedInputStream 缓冲
读写对象
OutputStream
FileOutputStream 文件
ByteArrayOutputStream 数组
ObjectOutputStream 对象
带缓冲读写
读写基本类型数据
FilterOutputStream
BufferedOutputStream 缓冲 flush()
DataOutputStream 基本数据类型 固定使用UTF-8编码
PrintStream
III. 字符流
以字符为单位进行读写（16位-Unicode编码）
Reader
FileReader 文件 默认采用平台编码，不能指定编码
CharArrayReader 数组
StringReader 字符串
InputStreamReader 转换流 InputStream->Reader 编码转换
BufferedReader 缓冲 readLine()
Writer
FileWriter 文件 默认采用平台编码，不能指定编码
CharArrayWriter 数组
StringWriter 字符串
OutputStreamWriter 转换流 OutputStream->Writer 编码转换
BufferedWriter 缓冲
PrintWriter
包装 Writer OutputStream
方法 println() 自动刷新 print()
按行读/写
IV. 案例
【例12-1】实现Windows环境下的dir命令 File
【例12-2】利用RandomAccessFile向文件写入一个字符和一个int型数据 RandomAccessFile 文件指针
【例12-3】向文件写入10个随机整数，并打印 RandomAccessFile
【例12-4】实现文件的多线程下载 RandomAccessFile 多线程
【例12-5】加密复制文件 文件读写 FileInputStream FileOutputStream
【例12-6】对比带缓冲的复制与不带缓冲复制的性能 缓冲读写 BufferedInputStream BufferedOutputStream
【例12-7】将1000~2000间的素数写入文件，并打印输出 基本数据类型读写 DataInputStream DataOutputStream
【例12-8】将一个Student对象持久化到文件，并读出打印 对象读写 Serializable ObjectInputStream ObjectOutputStream
【例12-9】实现对象的深复制 Serializable ObjectInputStream ByteInputStream ObjectOutputStream ByteOutputStream
【例12-10】向文件中写入一个中文字符串，并读出 中文编码 字符串读写 InputStreamReader(FileInputStream) OutputStreamWriter(FileOutputStream)
【例12-11】将九九乘法表保存在文本文件中 默认字符集的字符/字符串读写 FileReader FileWriter
【例12-12】将计算得到的几个圆面积写入文件，并读出打印 按行读写 FileOutputStream PrintWriter