IO流(二)

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

今天同学们需要学会什么

字符流、缓冲流	转换流	序列化	打印流	IO框架等
字符流更适合读取文本数据；之前的流性能偏弱，今天要学习读写数据的性能更好的流：缓冲流	使用字符流读取中文不会乱码，原因是什么？那么如果读取的文件编码与代码编码不一致怎么办？	如何把Java对象进行长久保存。	开发中有一种使用极为方便，性能高效的写数据的流，使用的很多。	IO流原生的API使用起来其实挺麻烦的，有没有更好用的方式。

目录

Contents

- 字符流
  - 字符输入流- 一次读取一个字符
  - 字符输入流- 一次读取一个字符数组
  - 字符输出流
- 缓冲流
- 转换流
- 序列化对象
- 打印流
- 补充知识：Properties
- 补充知识： IO框架

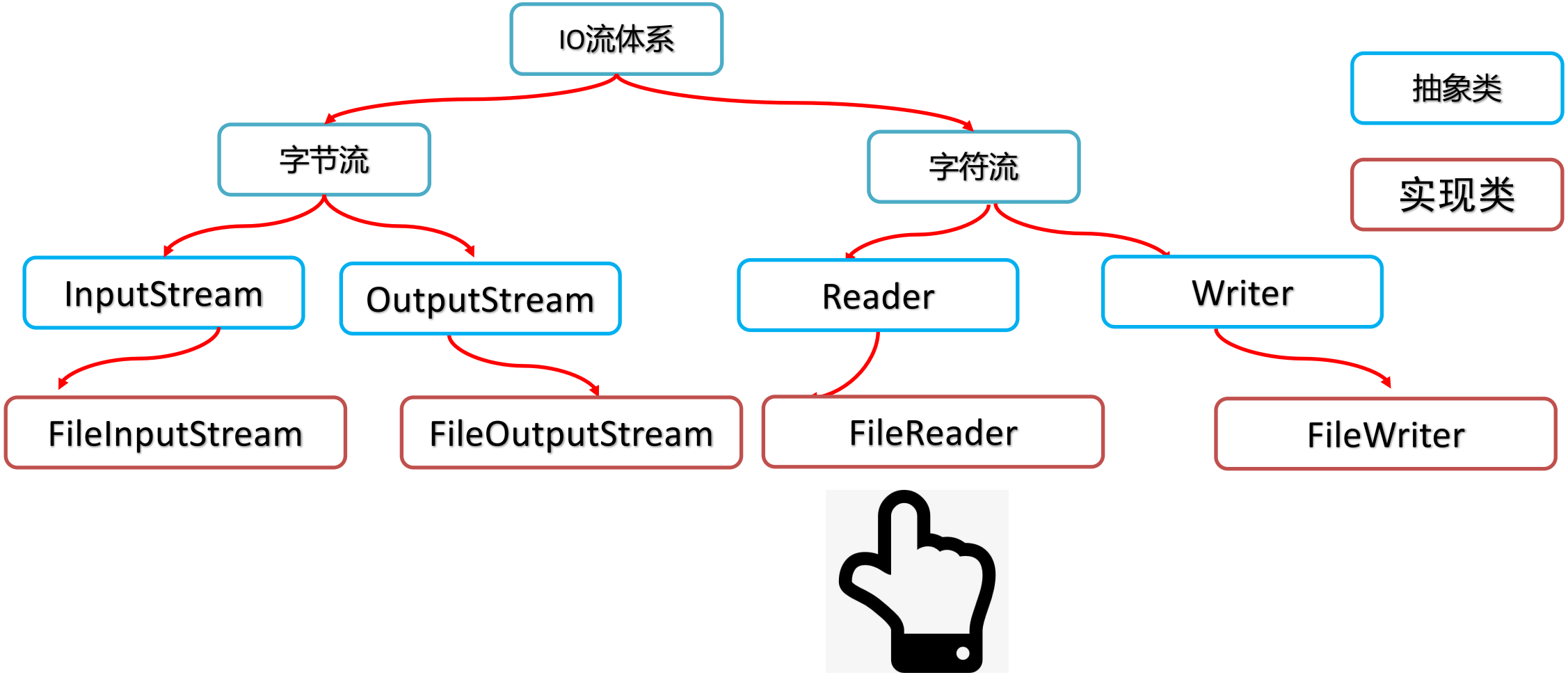
1. 字节流读取中文输出会存在什么问题？
   - **会乱码。或者内存溢出。**
2. 读取中文输出，哪个流更合适，为什么？
   - **字符流更合适，最小单位是按照单个字符读取的。**

磁盘

文件字符输入流，一个一个字符读数据到内存

内存

程序
进程

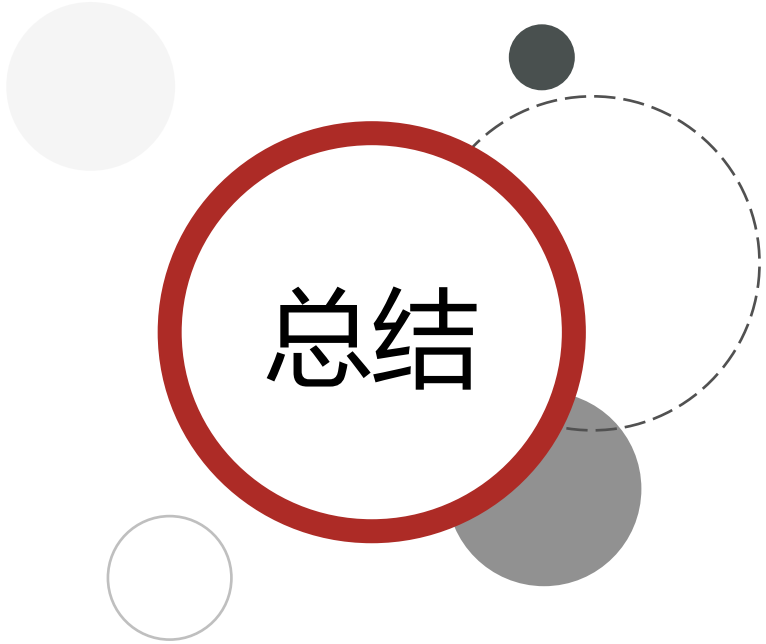

IO流体系
字节流
字符流
抽象类
实现类
InputStream
OutputStream
Reader
Writer
FileInputStream
FileOutputStream
FileReader
FileWriter

文件字符输入流：Reader

- 作用：以内存为基准，把磁盘文件中的数据以字符的形式读取到内存中去。

构造器	说明
public FileReader(File file)	创建字符输入流管道与源文件对象接通
public FileReader(String pathname)	创建字符输入流管道与源文件路径接通

方法名称	说明
public int read()	每次读取一个字符返回，如果字符已经没有可读的返回-1
public int read(char[] buffer)	每次读取一个字符数组，返回读取的字符个数，如果字符已经没有可读的返回-1



1. 文件字符输入流，每次读取一个字符的api是哪个？

方法名称	说明
public int read()	每次读取一个字符返回，如果字节已经没有可读的返回-1

2. 字符流的好处。每次读取一个字符存在什么问题？
   - **读取中文字符不会出现乱码（如果代码文件编码一致）**
   - **性能较慢**

目录

Contents

- **字符流的使用**
  - 字符输入流- 一次读取一个字符
  - 字符输入流- 一次读取一个字符数组
  - 字符输出流
- **缓冲流**
- **转换流**
- **序列化对象**
- **打印流**
- **补充知识：Properties**
- **补充知识： IO框架**

磁盘

文件字节输符流，一个个字符数组读数据到内存

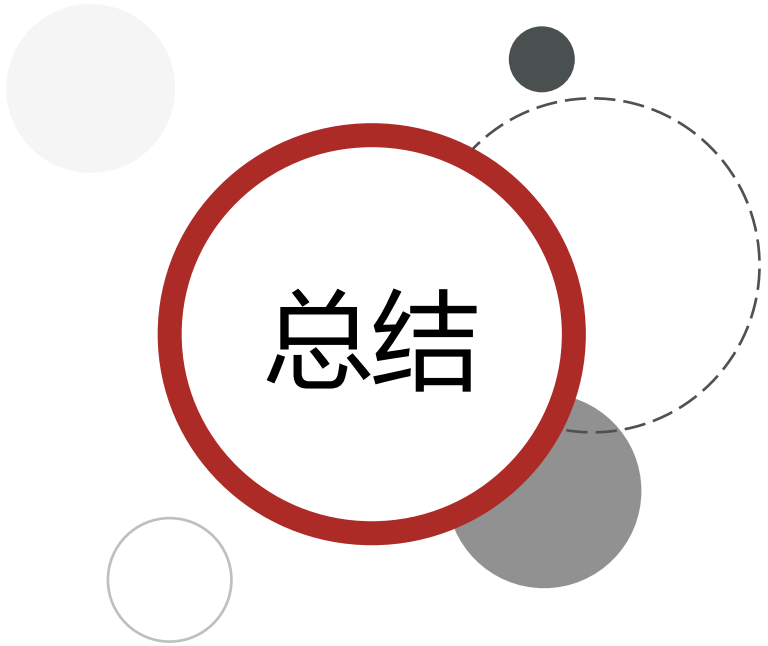
内存

程序
进程

文件字符输入流：FileReader

- 作用：以内存为基准，把磁盘文件中的数据以字符的形式读取到内存中去。

方法名称	说明
public int read()	每次读取一个字符返回，如果字符已经没有可读的返回-1
public int read(char[] buffer)	每次读取一个字符数组，返回读取的字符数，如果字符已经没有可读的返回-1



1. 文件字符输入流，每次读取一个字符数组的api是哪个？

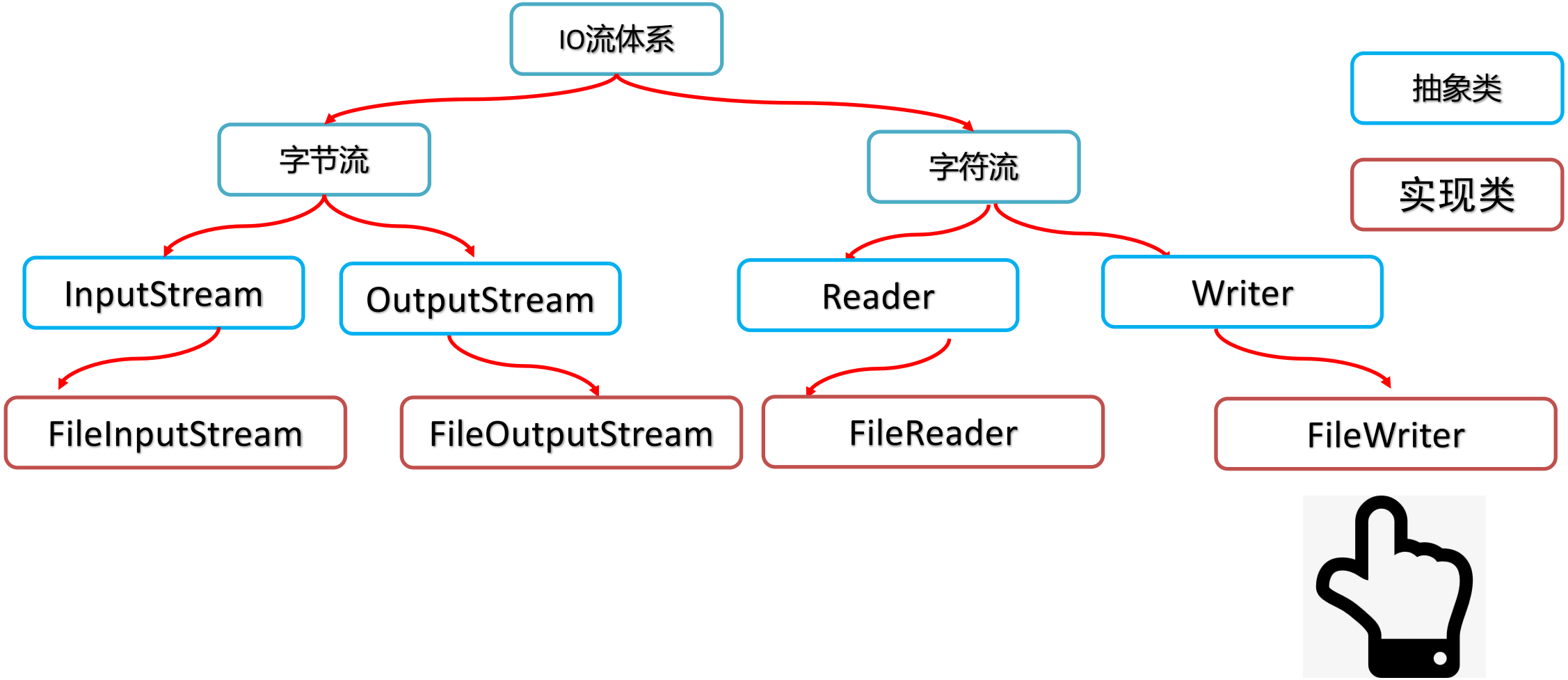
方法名称	说明
public int read(char[] buffer)	每次读取一个字符数组，返回读取的字符个数，如果字符已经没有可读的返回-1

2. 每次读取一个字符数组的优势？
   - **读取的性能得到了提升**
   - **读取中文字符输出不会乱码。**

目录

Contents

- **字符流的使用**
  - 字符输入流- 一次读取一个字符
  - 字符输入流- 一次读取一个字符数组
  - 字符输出流
- **缓冲流**
- **转换流**
- **序列化对象**
- **打印流**
- **补充知识：Properties**
- **补充知识： IO框架**


IO流体系
字节流
字符流
抽象类
实现类
InputStream
OutputStream
Reader
Writer
FileInputStream
FileOutputStream
FileReader
FileWriter

磁盘

文件字节输出流，写字符读数据到磁盘文件

内存

程序
进程

文件字符输出流：FileWriter

- 作用：以内存为基准，把内存中的数据以字符的形式写出到磁盘文件中去的流。

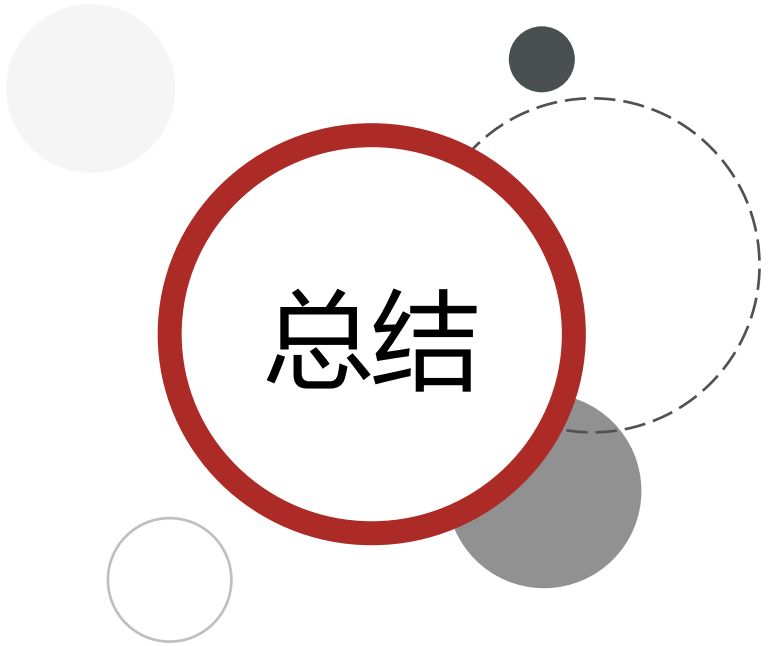
构造器	说明
public FileWriter(File file)	创建字符输出流管道与源文件对象接通
public FileWriter (File file , boolean append)	创建字符输出流管道与源文件对象接通，可追加数据
public FileWriter (String filepath)	创建字符输出流管道与源文件路径接通
public FileWriter (String filepath , boolean append)	创建字符输出流管道与源文件路径接通，可追加数据

文件字符输出流（FileWriter）写数据出去的API

方法名称	说明
void write(int c)	写一个字符
void write(char[] cbuf)	写入一个字符数组
void write(char[] cbuf, int off, int len)	写入字符数组的一部分
void write(String str)	写一个字符串
void write(String str, int off, int len)	写一个字符串的一部分

流的关闭与刷新

方法	说明
flush()	刷新流，还可以继续写数据
close()	关闭流，释放资源，但是在关闭之前会先刷新流。一旦关闭，就不能再写数据



1. 字符输出流写数据的方法有哪些

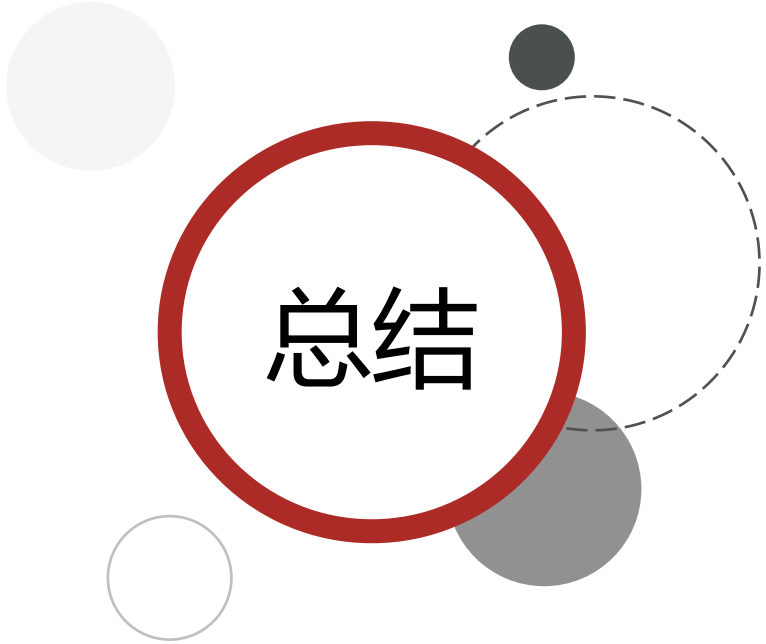
方法名称	说明
void write(int c)	写一个字符
void write(char[] cbuf)	写入一个字符数组
void write(char[] cbuf, int off, int len)	写入字符数组的一部分
void write(String str)	写一个字符串
void write(String str, int off, int len)	写一个字符串的一部分

2. 字符输出流如何实现数据追加

`public FileWriter (String filepath , boolean append)`	创建字符输出流管道与源文件路径接通，可追加数据

总结

3. 字符输出流如何实现写出去的数据能换行
   - **fw.write(“\r\n”)**
4. 字符输出流如何实现写出去的数据能换行
   - **flush()刷新数据**
   - **close()方法是关闭流，关闭包含刷新，关闭后流不可以继续使用了。**



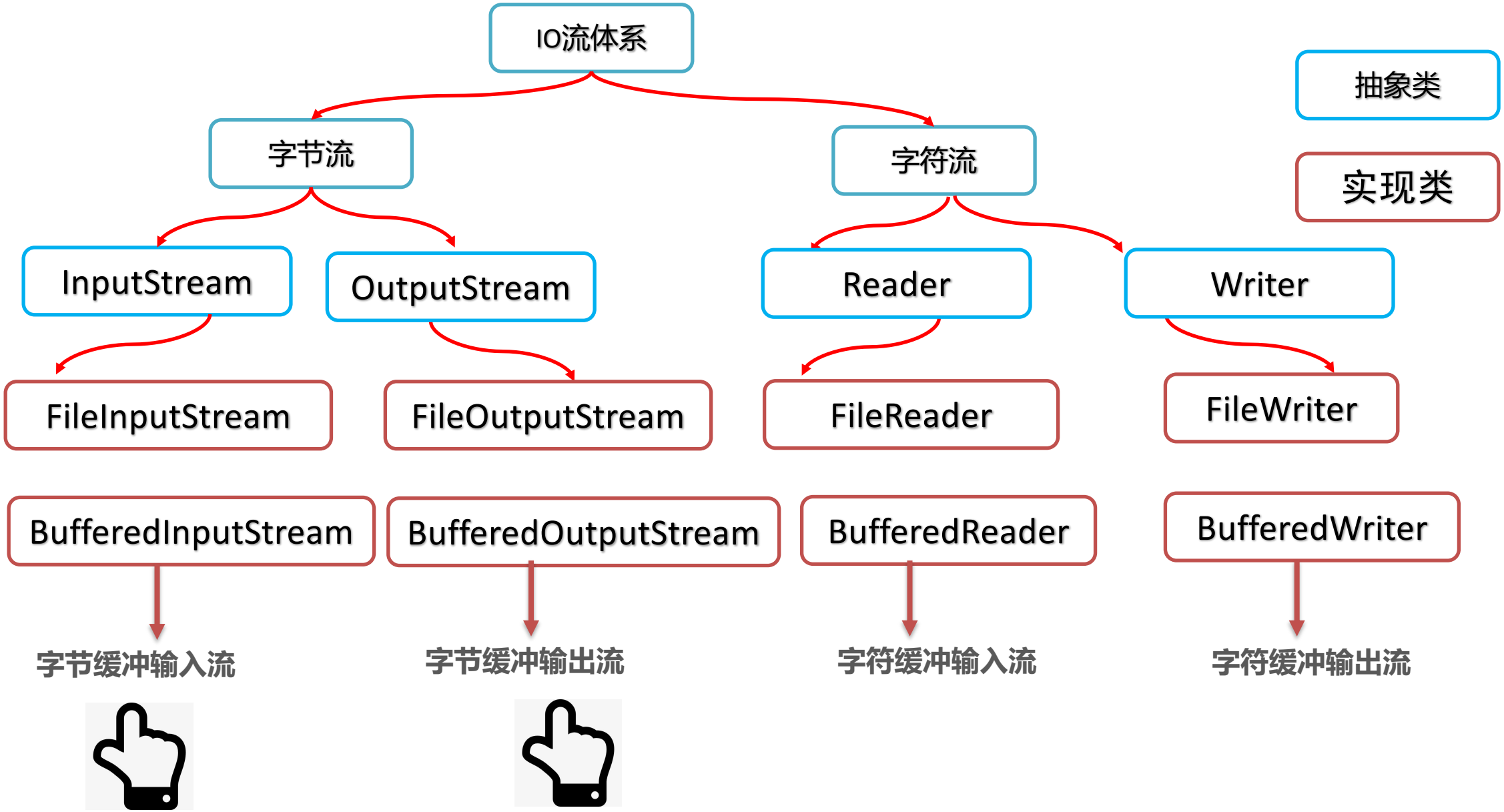
5. 字节流、字符流的使用场景总结？

- **字节流适合做一切文件数据的拷贝（音视频，文本）**
- **字节流不适合读取中文内容输出**
- **字符流适合做文本文件的操作（读，写）**

目录

Contents

- 字符流
- 缓冲流
  - 缓冲流概述、字节缓冲流的使用
  - 字节缓冲流的性能分析
  - 字符缓冲流
- 转换流
- 序列化对象
- 打印流
- 补充知识：Properties
- 补充知识： IO框架


IO流体系
字节流
字符流
抽象类
实现类
InputStream
OutputStream
Reader
Writer
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
字节缓冲输入流
字节缓冲输出流
字符缓冲输入流
字符缓冲输出流

缓冲流概述

- 缓冲流也称为高效流、或者高级流。之前学习的字节流可以称为原始流。
- 作用：缓冲流自带缓冲区、可以提高原始字节流、字符流读写数据的性能

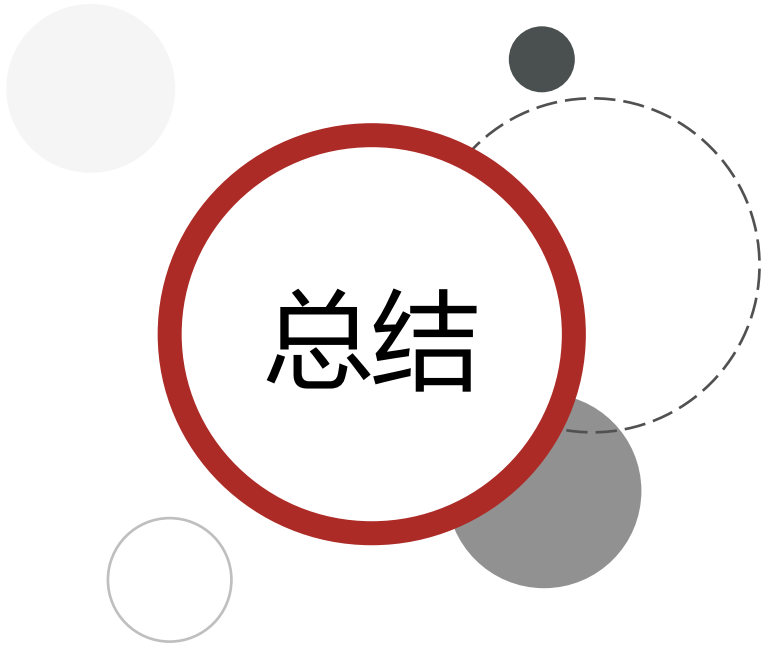
字节缓冲流性能优化原理：

- 字节缓冲输入流自带了8KB缓冲池，以后我们直接从缓冲池读取数据，所以性能较好。
- 字节缓冲输出流自带了8KB缓冲池，数据就直接写入到缓冲池中去，写数据性能极高了。

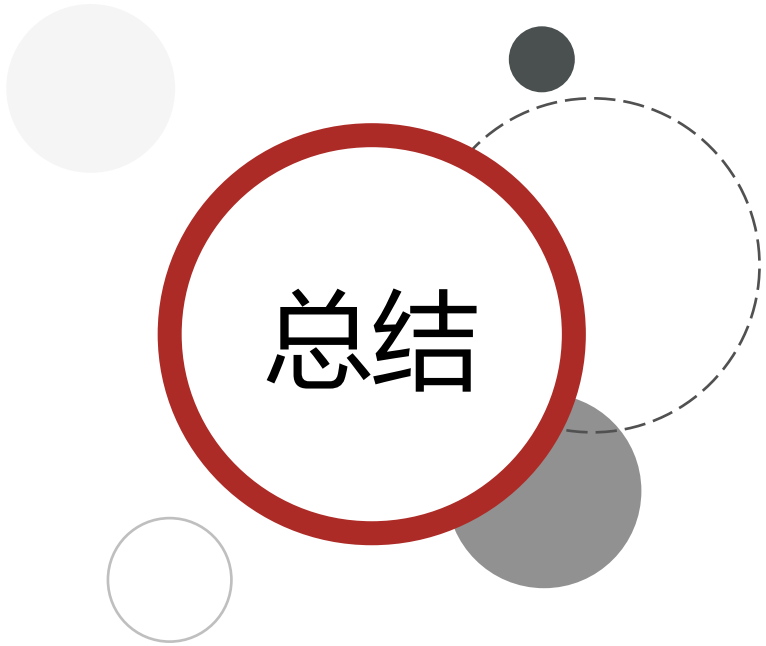
字节缓冲流

- 字节缓冲输入流：BufferedInputStream，提高字节输入流读取数据的性能。
- 字节缓冲输出流：BufferedOutputStream，提高字节输出流读取数据的性能。

构造器	说明
public BufferedInputStream(InputStream is)	可以把低级的字节输入流包装成一个高级的缓冲字节输入流管道，从而提高字节输入流读数据的性能
public BufferedOutputStream(OutputStream os)	可以把低级的字节输出流包装成一个高级的缓冲字节输出流，从而提高写数据的性能



1. 缓冲流的作用？
   - **缓冲流自带缓冲区、可以提高原始字节流、字符流读写数据的性能**
2. 缓冲流有几种？
   - **字节缓冲流**
     - **字节缓冲输入流： BufferedInputStream**
     - **字节缓冲输出流：BufferedOutputStream**
   - **字符缓冲流**
     - **字符缓冲输入流：BufferedReader**
     - **字符缓冲输出流：BufferedWriter**



1. 字节缓冲流为什么提高了操作数据的性能？
   - **字节缓冲流自带8KB缓冲区**
   - **可以提高原始字节流、字符流读写数据的性能**
2. 字节缓冲流的功能如何调用？
   - **public BufferedOutputStream(OutputStream os)**
   - **public BufferedInputStream(InputStream is)**
   - **功能上并无很大变化，性能提升了。**

目录

Contents

- 字符流
- 缓冲流
  - 缓冲流概述、字节缓冲流的使用
  - 字节缓冲流的性能分析
  - 字符缓冲流
- 转换流
- 序列化对象
- 打印流
- 补充知识：Properties
- 补充知识： IO框架

1、我们已经说明了字节缓冲流的性能高效，但是没有直接感受到。

2、如何测试字节缓冲流的读写性能呢？

步骤 分别使用不同的方式复制大视频观察性能情况

需求

- 分别使用低级字节流和高级字节缓冲流拷贝大视频，记录耗时。

分析

① 使用低级的字节流按照一个一个字节的形式复制文件。

② 使用低级的字节流按照一个一个字节数组的形式复制文件。

③ 使用高级的缓冲字节流按照一个一个字节的形式复制文件。

④ 使用高级的缓冲字节流按照一个一个字节数组的形式复制文件。

总结

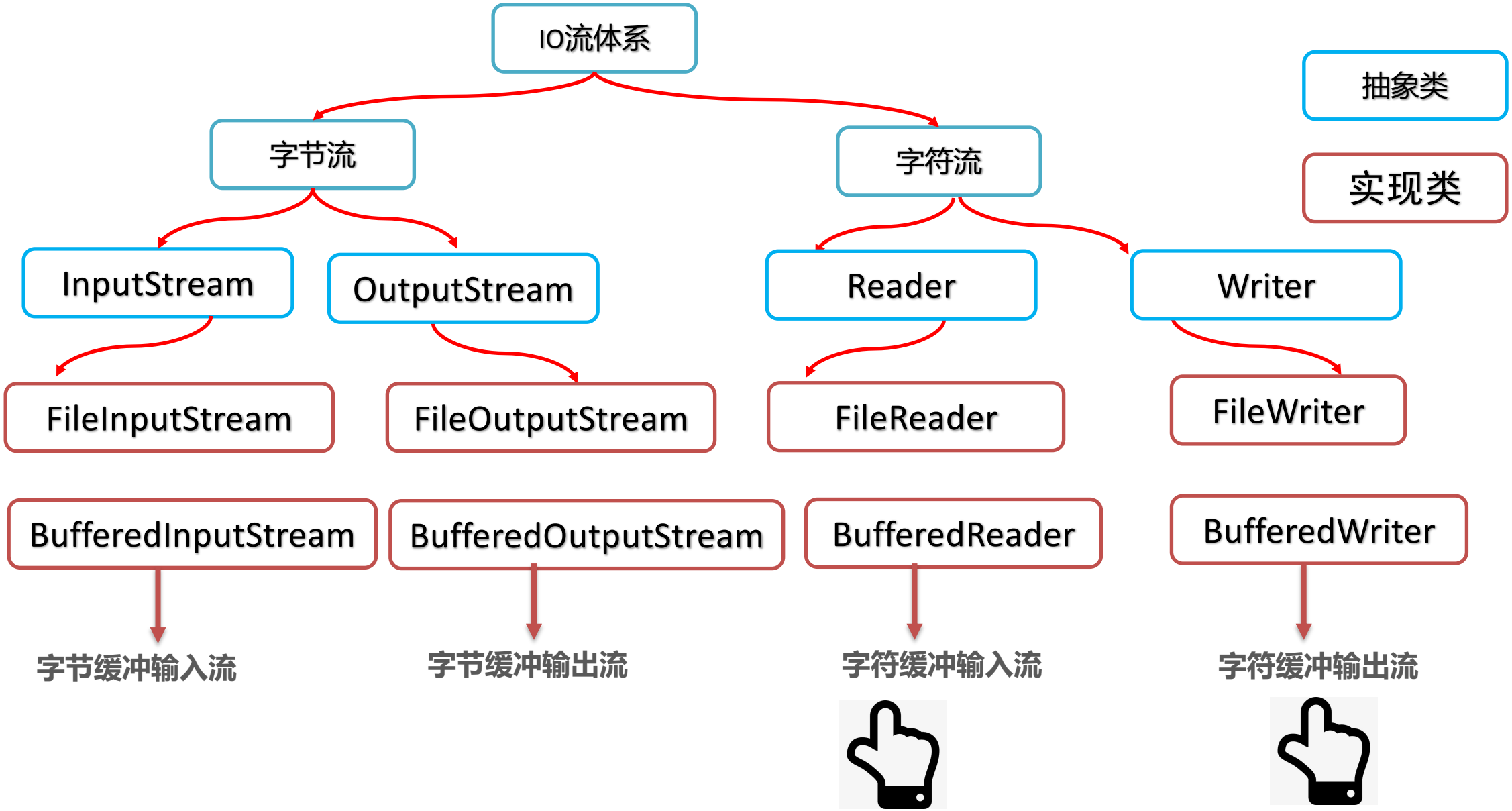
1. 推荐使用哪种方式提高字节流读写数据的性能？

   - **建议使用字节缓冲输入流、字节缓冲输出流，结合字节数组的方式，目前来看是性能最优的组合。**

目录

Contents

- 字符流
- 缓冲流
  - 缓冲流概述、字节缓冲流的使用
  - 字节缓冲流的性能分析
  - 字符缓冲流
- 转换流
- 序列化对象
- 打印流
- 补充知识：Properties
- 补充知识： IO框架


IO流体系
字节流
字符流
抽象类
实现类
InputStream
OutputStream
Reader
Writer
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
字节缓冲输入流
字节缓冲输出流
字符缓冲输入流
字符缓冲输出流

字符缓冲输入流

- 字符缓冲输入流：BufferedReader。
- 作用：提高字符输入流读取数据的性能，除此之外多了按照行读取数据的功能。

构造器	说明
`public BufferedReader (Reader r)`	可以把低级的字符输入流包装成一个高级的缓冲字符输入流管道，从而提高字符输入流读数据的性能

字符缓冲输入流新增功能

方法	说明
`public String readLine()`	读取一行数据返回，如果读取没有完毕，无行可读返回null

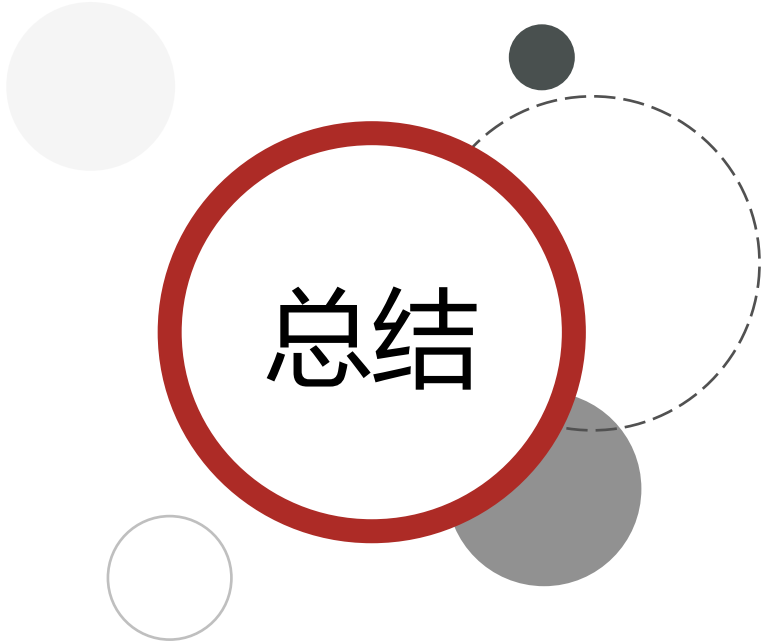
字符缓冲输出流

- 字符缓冲输出流：BufferedWriter。
- 作用：提高字符输出流写取数据的性能，除此之外多了换行功能

构造器	说明
public BufferedWriter(Writer w)	可以把低级的字符输出流包装成一个高级的缓冲字符输出流管道，从而提高字符输出流写数据的性能

字符缓冲输出流新增功能

方法	说明
`public void newLine()`	换行操作



1. 字符缓冲流为什么提高了操作数据的性能？

- **字符缓冲流自带8K缓冲区**
- **可以提高原始字符流读写数据的性能**

2. 字符缓冲流的功能如何使用？

- **public BufferedReader(Reader r)**
- **性能提升了,多了readLine()按照行读取的功能**
- **public BufferedWriter(Writer w)**
- **性能提升了,多了newLine()换行的功能**

案例

拷贝出师表到另一个文件，恢复顺序

需求：把《出师表》的文章顺序进行恢复到一个新文件中。

分析：

① 定义一个缓存字符输入流管道与源文件接通。

② 定义一个List集合存储读取的每行数据。

③ 定义一个循环按照行读取数据，存入到List集合中去。

④ 对List集合中的每行数据按照首字符编号升序排序。

⑤ 定义一个缓存字符输出管道与目标文件接通。

⑥ 遍历List集合中的每个元素，用缓冲输出管道写出并换行。

目录

Contents

- 字符流
- 缓冲流
- 转换流
  - 问题引出：不同编码读取乱码问题
  - 字符输入转换流
  - 字符输出转换流
- 序列化对象
- 打印流
- Properties
- IO框架

1、之前我们使用字符流读取中文是否有乱码？

- **没有的，因为代码编码和文件编码都是UTF-8。**

2、如果代码编码和文件编码不一致，使用字符流直接读取还能不乱码吗？

- **会乱码。**
- **文件编码和读取的编码必须一致才不会乱码。**

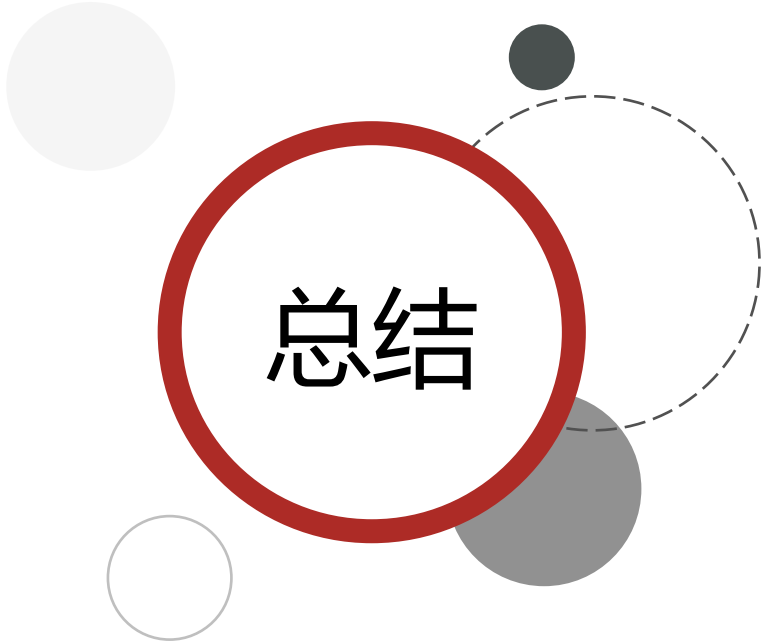
步骤

使用相同编码读取不同编码的文件内容

需求：分别使用如下两种方式读取文件内容

① 代码编码是UTF-8，文件编码也是UTF-8，使用字符流读取观察输出的中文字符结果。

② 代码编码是UTF-8，文件编码使用GBK，使用字符流读取观察输出的中文字符结果



1. 字符流直接读取文本内容。

- **必须文件和代码编码一致才不会乱码**
- **如果文件和代码编码不一致，读取将出现乱码。**

目录

Contents

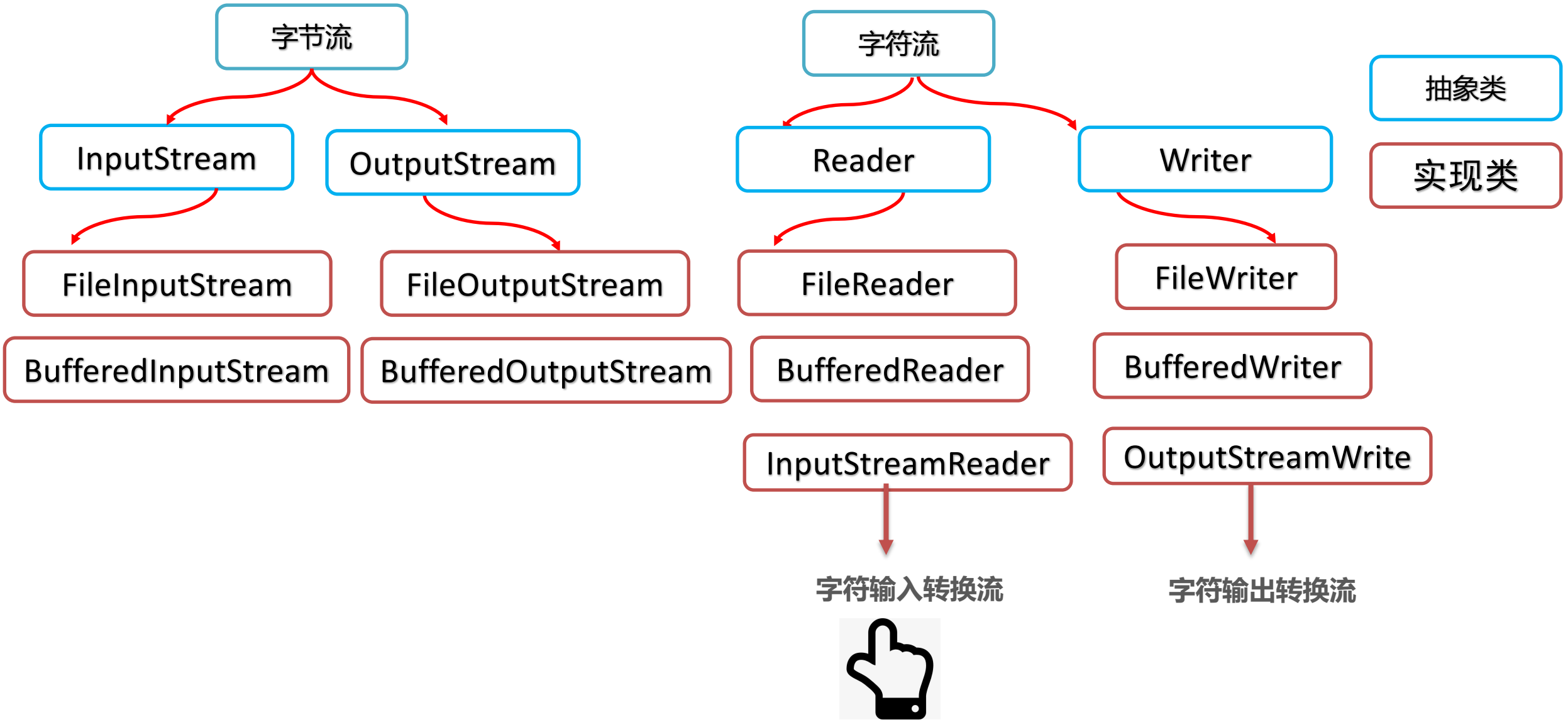
- 缓冲流
- 转换流
  - 问题引出：不同编码读取乱码问题
  - 字符输入转换流
  - 字符输出转换流
- 序列化对象
- 打印流
- Properties
- IO框架

1、如果代码编码和文件编码不一致，使用字符流直接读取还能不乱码吗？

- **会乱码。**

2、如果如何解决呢？

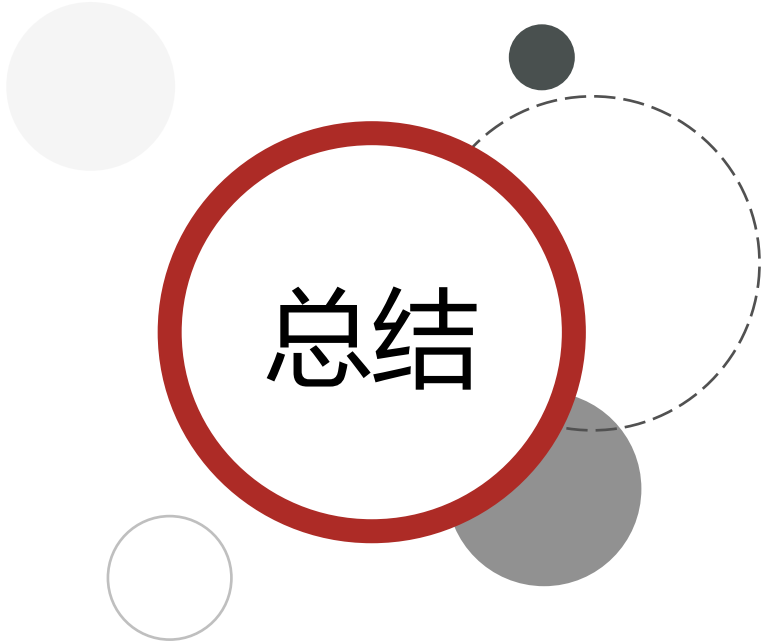
- **使用字符输入转换流**
- **可以提取文件（GBK）的原始字节流，原始字节不会存在问题。**
- **然后把字节流以指定编码转换成字符输入流，这样字符输入流中的字符就不乱码了**


字节流
字符流
抽象类
实现类
InputStream
OutputStream
Reader
Writer
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
InputStreamReader
OutputStreamWrite
字符输入转换流
字符输出转换流

字符输入转换流

- 字符输入转换流：InputStreamReader，可以把原始的字节流按照指定编码转换成字符输入流。

构造器	说明
public InputStreamReader(InputStream is)	可以把原始的字节流按照代码默认编码转换成字符输入流。几乎不用，与默认的FileReader一样。
public InputStreamReader(InputStream is ，String charset)	可以把原始的字节流按照指定编码转换成字符输入流，这样字符流中的字符就不乱码了(重点)



1. 字符输入转换流InputStreamReader作用：

   - **可以解决字符流读取不同编码乱码的问题**
   - **public InputStreamReader(InputStream is,String charset):**

     可以指定编码把原始字节流转换成字符流，如此字符流中的字符不乱码。

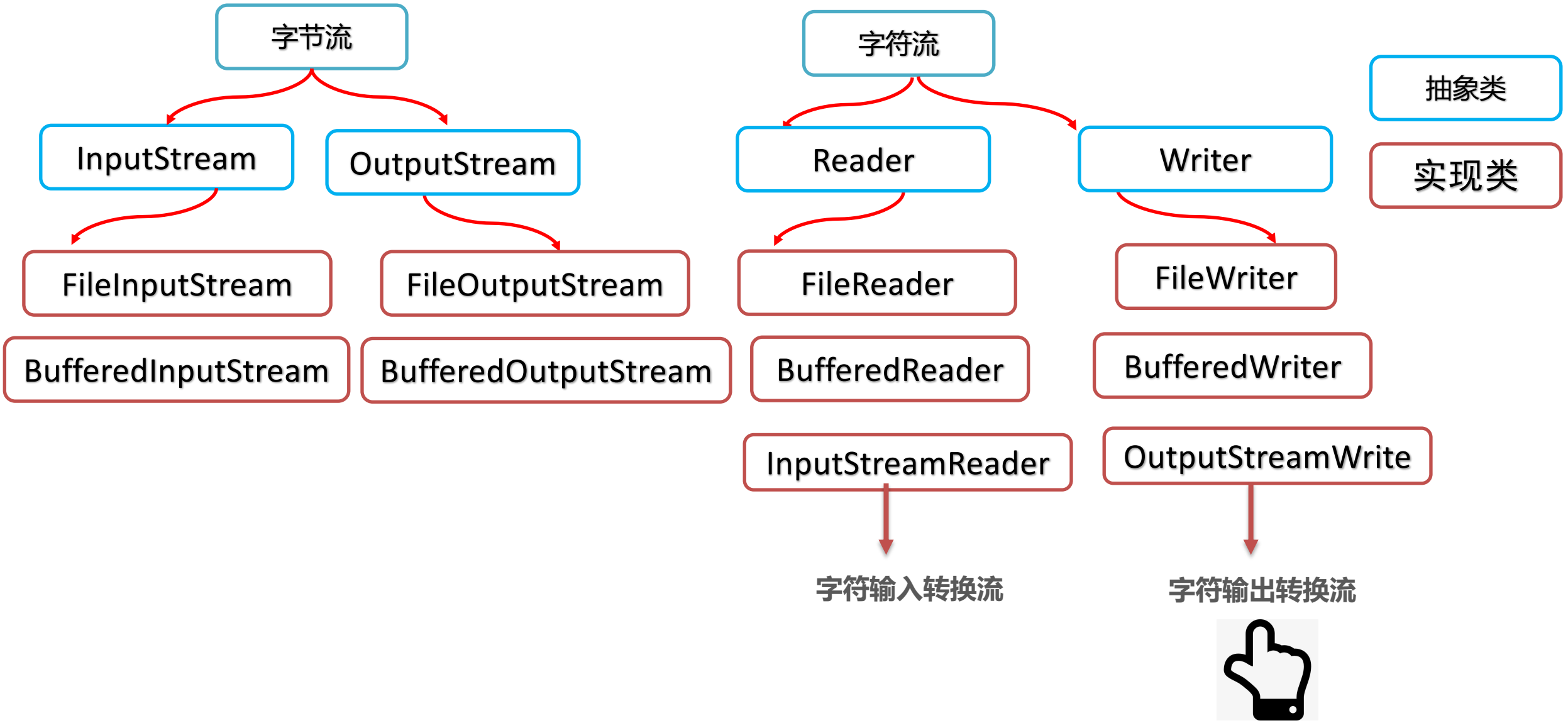
目录

Contents

- 缓冲流
- 转换流
  - 问题引出：不同编码读取乱码问题
  - 字符输入转换流
  - 字符输出转换流
- 序列化对象
- 打印流
- 补充知识：Properties
- 补充知识：IO框架

1、如果需要控制写出去的字符使用的编码，怎么办？

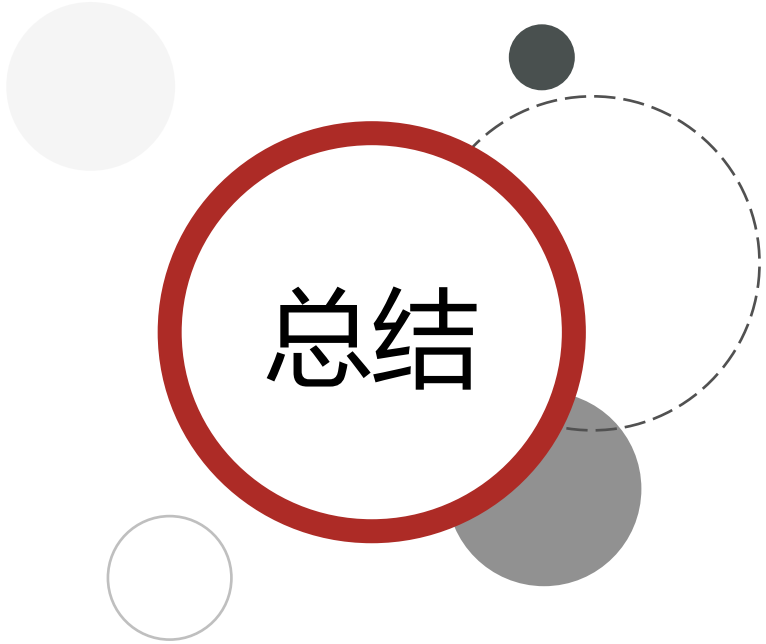
- **可以把字符以指定编码获取字节后再使用字节输出流写出去：**
  - **“我爱你中国”.getBytes(编码)**
- **也可以使用字符输出转换流实现。**


字节流
字符流
抽象类
实现类
InputStream
OutputStream
Reader
Writer
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
InputStreamReader
OutputStreamWrite
字符输入转换流
字符输出转换流

字符输出转换流

- 字符输入转换流：OutputStreamWriter，可以把字节输出流按照指定编码转换成字符输出流。

构造器	说明
public OutputStreamWriter(OutputStream os)	可以把原始的字节输出流按照代码默认编码转换成字符输出流。几乎不用。
public OutputStreamWriter(OutputStream os，String charset)	可以把原始的字节输出流按照指定编码转换成字符输出流(重点)



1. 字符输出转换流OutputStreamWriter的作用？
   - **public OutputStreamWriter(OutputStream os，String charset)**
   - **可以指定编码把字节输出流转换成字符输出流，从而可以指定写出去的字符编码！**

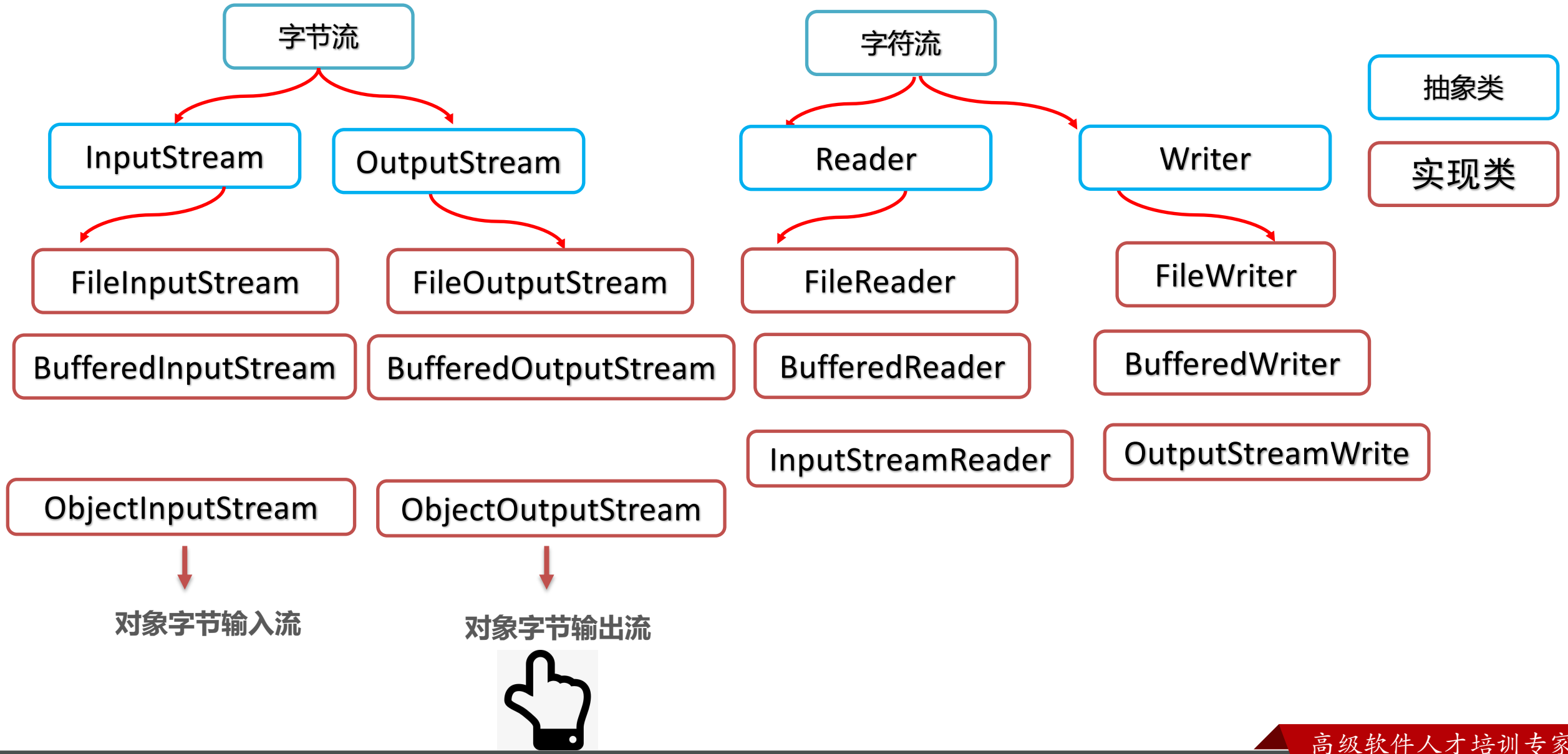
目录

Contents

- 缓冲流
- 转换流
- 序列化对象
  - 对象序列化
  - 对象反序列化
- 打印流
- 补充知识：Properties
- 补充知识：IO框架

对象序列化：

- 作用：以内存为基准，把内存中的对象存储到磁盘文件中去，称为对象序列化。
- 使用到的流是对象字节输出流：ObjectOutputStream


字节流
字符流
抽象类
InputStream
OutputStream
Reader
Writer
实现类
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
InputStreamReader
OutputStreamWrite
ObjectInputStream
ObjectOutputStream
对象字节输入流
对象字节输出流

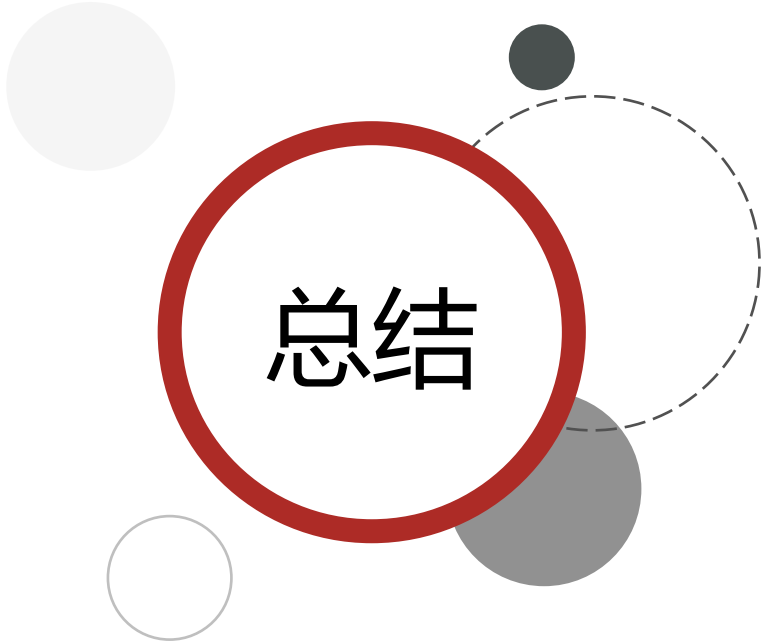
对象序列化：

- 使用到的流是对象字节输出流：ObjectOutputStream

构造器	说明
public ObjectOutputStream(OutputStream out)	把低级字节输出流包装成高级的对象字节输出流

ObjectOutputStream序列化方法

方法名称	说明
public final void writeObject(Object obj)	把对象写出去到对象序列化流的文件中去

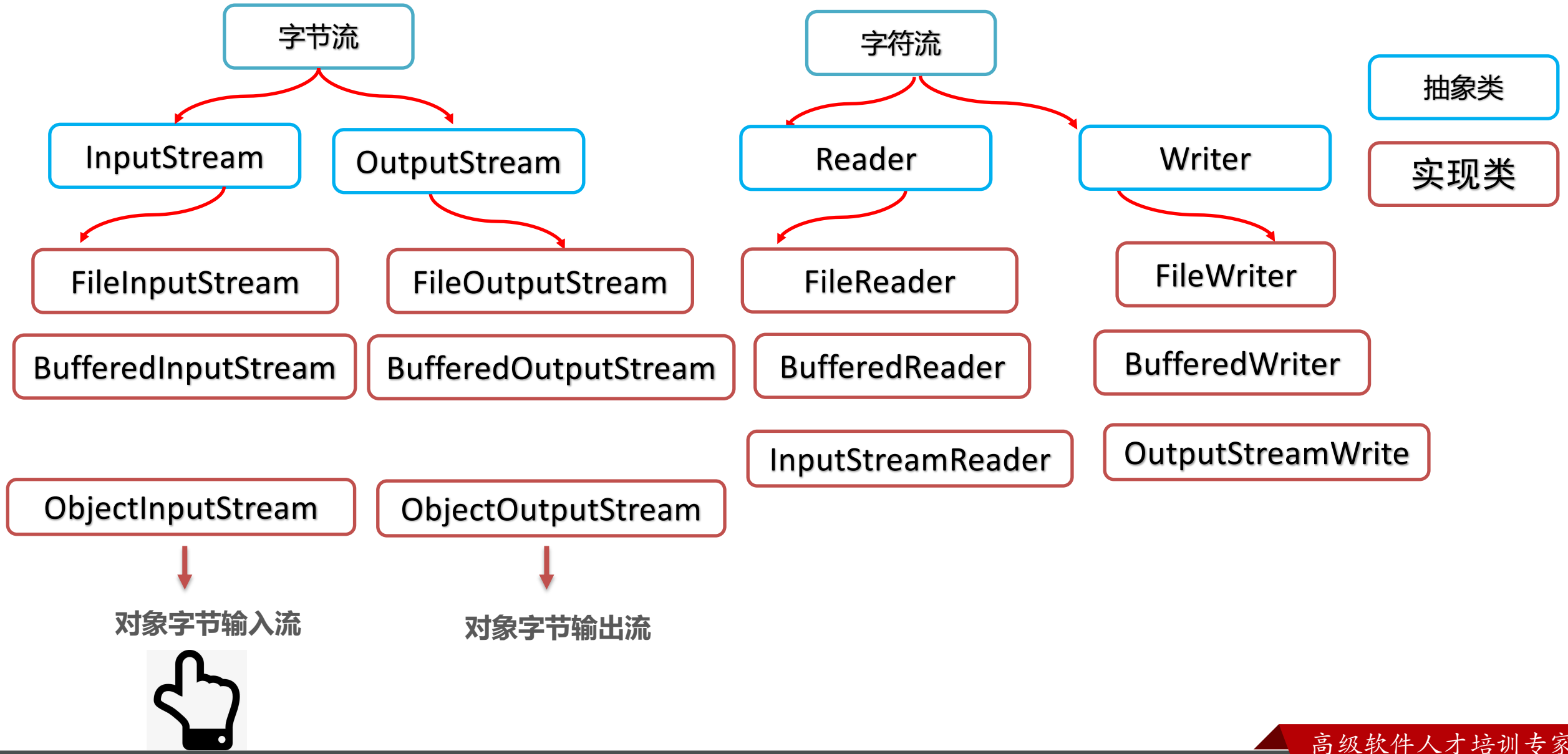


1. 对象序列化的含义是什么？
   - **把对象数据存入到文件中去。**
2. 对象序列化用到了哪个流？
   - **对象字节输出流ObjectOutputStram**
   - **public void writeObject(Object obj)**
3. 序列化对象的要求是怎么样的？
   - **对象必须实现序列化接口**

目录

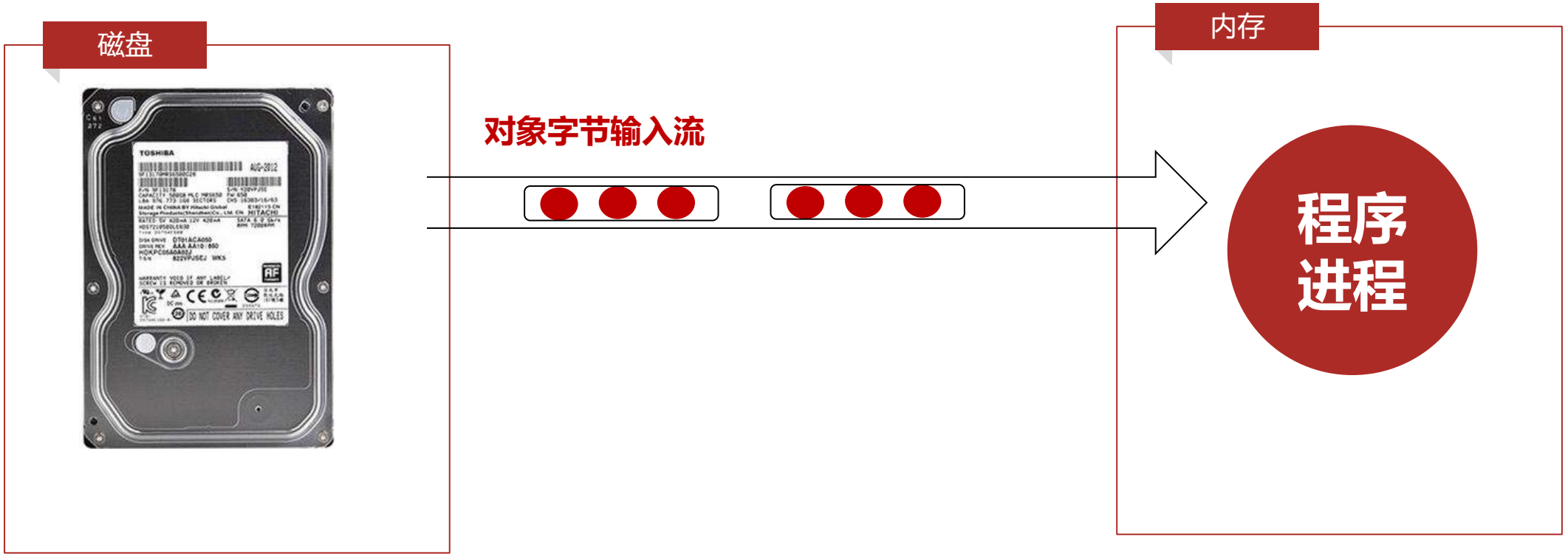
Contents

- 缓冲流
- 转换流
- 序列化对象
  - 对象序列化
  - 对象反序列化
- 打印流
- 补充知识：Properties
- 补充知识：IO框架


字节流
字符流
抽象类
InputStream
OutputStream
Reader
Writer
实现类
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
InputStreamReader
OutputStreamWrite
ObjectInputStream
ObjectOutputStream
对象字节输入流
对象字节输出流

对象反序列化：

- 使用到的流是对象字节输入流：ObjectInputStream
- 作用：以内存为基准，把存储到磁盘文件中去的对象数据恢复成内存中的对象，称为对象反序列化。



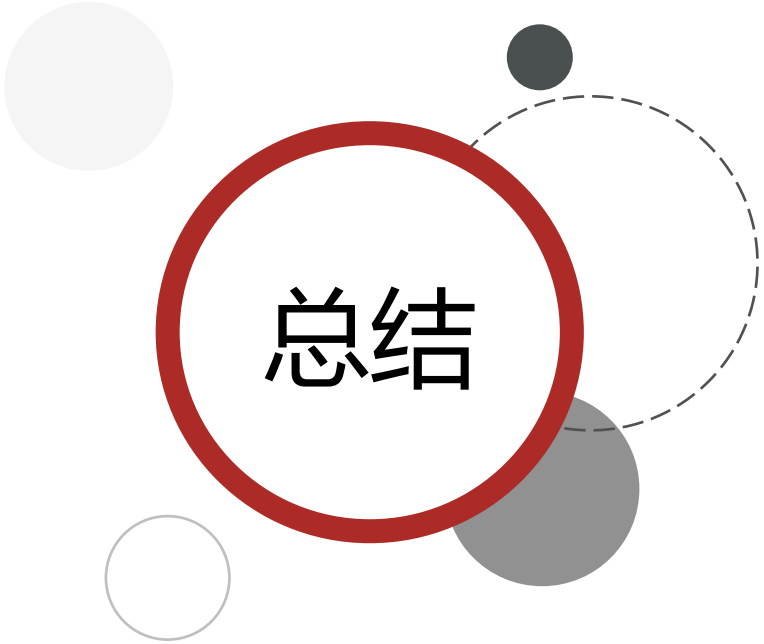
对象反序列化：

- 使用到的流是对象字节输入流：ObjectInputStream
- 作用：以内存为基准，把存储到磁盘文件中去的对象数据恢复成内存中的对象，称为对象反序列化。

构造器	说明
`public ObjectInputStream(InputStream out)`	把低级字节输如流包装成高级的对象字节输入流

ObjectInputStream序列化方法

方法名称	说明
public Object readObject()	把存储到磁盘文件中去的对象数据恢复成内存中的对象返回

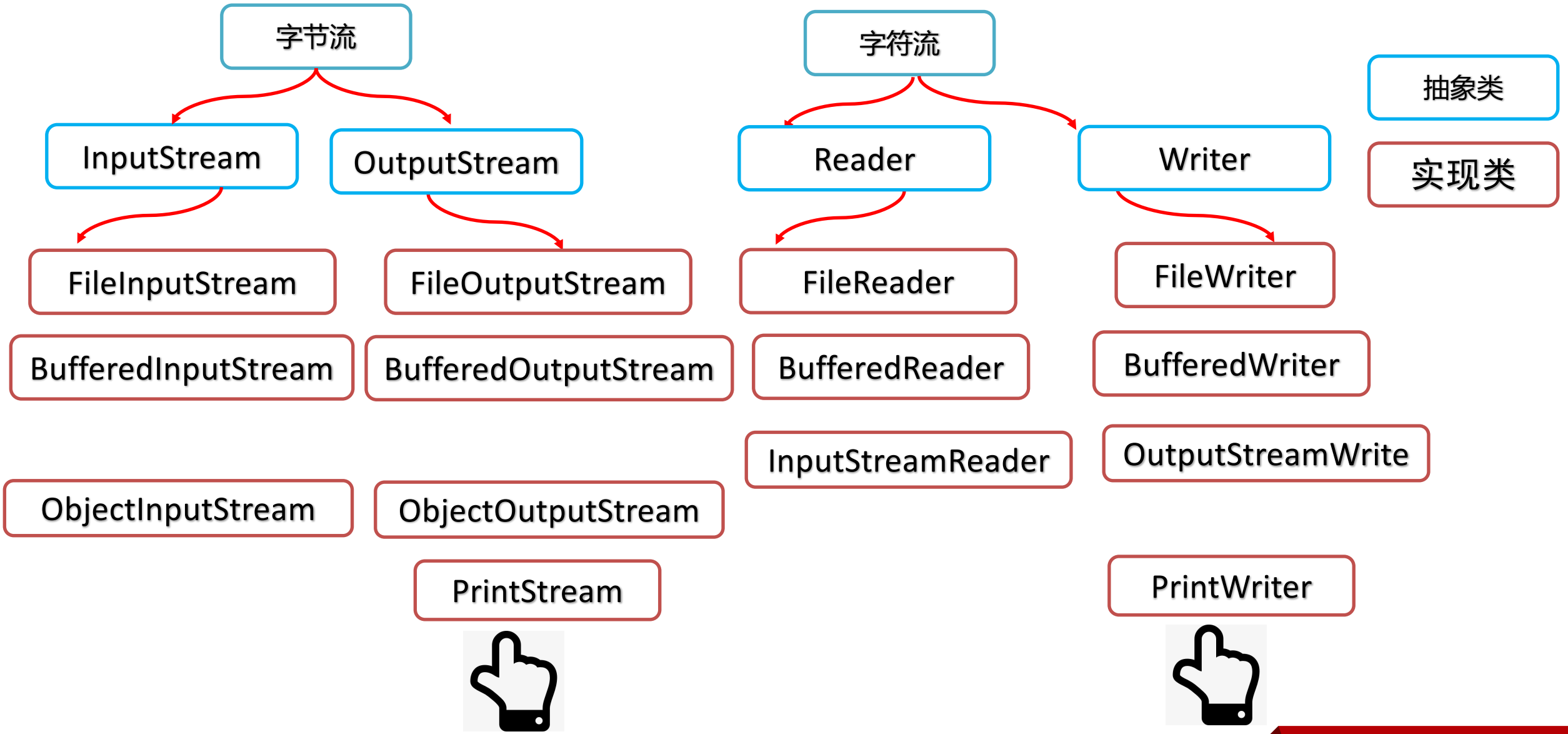


1. 对象反序列化的含义是什么？
    - **把磁盘中的对象数据恢复到内存的Java对象中。**
2. 对象反序列化用到了哪个流？
    - **对象字节输入流ObjectInputStram**
    - **public Object readObject()**

目录

Contents

- 缓冲流
- 转换流
- 序列化对象
- **打印流**
  - PrintStream、PrintWriter
  - 输出语句的重定向
- 补充知识：Properties
- 补充知识：IO框架


字节流
字符流
抽象类
InputStream
OutputStream
Reader
Writer
实现类
FileInputStream
FileOutputStream
FileReader
FileWriter
BufferedInputStream
BufferedOutputStream
BufferedReader
BufferedWriter
InputStreamReader
OutputStreamWrite
ObjectInputStream
ObjectOutputStream
PrintStream
PrintWriter

打印流

- 作用：打印流可以实现方便、高效的打印数据到文件中去。打印流一般是指：PrintStream，PrintWriter两个类。
- 可以实现打印什么数据就是什么数据，例如打印整数97写出去就是97，打印boolean的true，写出去就是true。

PrintStream

构造器	说明
`public PrintStream(OutputStream os)`	打印流直接通向字节输出流管道
`public PrintStream(File  f)`	打印流直接通向文件对象
`public PrintStream(String filepath)`	打印流直接通向文件路径

方法	说明
`public void print(Xxx xx)`	打印任意类型的数据出去

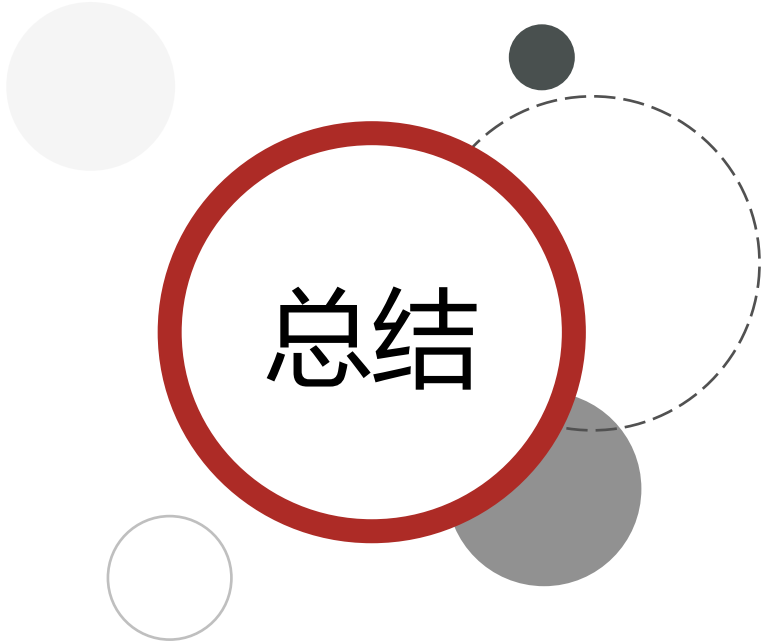
PrintWriter

构造器	说明
public PrintWriter(OutputStream os)	打印流直接通向字节输出流管道
public PrintWriter (Writer w)	打印流直接通向字符输出流管道
public PrintWriter (File f)	打印流直接通向文件对象
public PrintWriter (String filepath)	打印流直接通向文件路径

方法	说明
public void print(Xxx xx)	打印任意类型的数据出去

PrintStream和PrintWriter的区别

- 打印数据功能上是一模一样的，都是使用方便，性能高效（核心优势）
- PrintStream继承自字节输出流OutputStream，支持写字节数据的方法。
- PrintWriter继承自字符输出流Writer，支持写字符数据出去。



1. 打印流有几种？各有什么特点？
   - **打印流一般是指：PrintStream，PrintWriter两个类。**
   - **打印功能2者是一样的使用方式**
   - **PrintStream继承自字节输出流OutputStream，支持写字节**
   - **PrintWrite继承自字符输出流Writer，支持写字符**
2. 打印流的优势是什么？
   - **两者在打印功能上都是使用方便，性能高效（核心优势）**

目录

Contents

- 缓冲流
- 转换流
- 序列化对象
- 打印流
  - PrintStream、PrintWriter
  - 输出语句的重定向
- 补充知识：Properties
- 补充知识：IO框架

输出语句重定向

- 属于打印流的一种应用，可以把输出语句的打印位置改到文件。

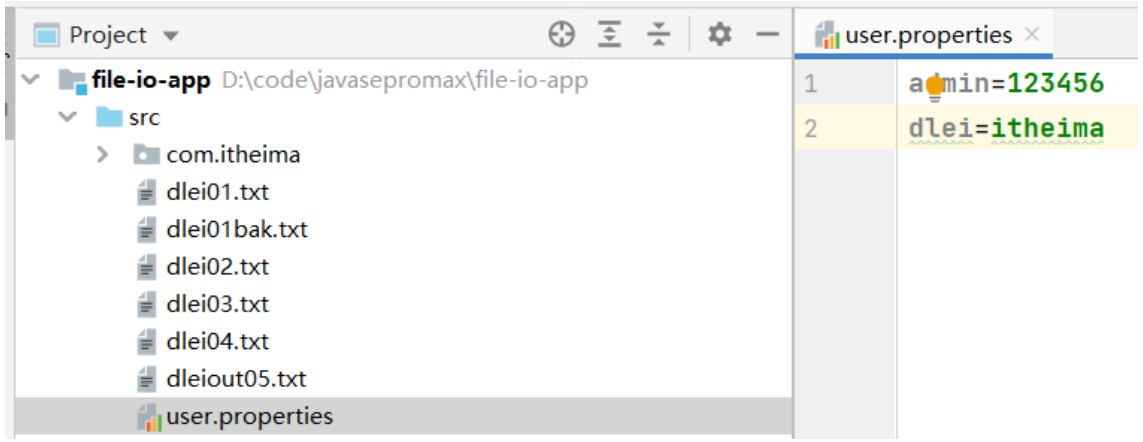
```
PrintStream ps = new PrintStream("文件地址")

System.setOut(ps);
```

目录

Contents

- 字符流
- 缓冲流
- 转换流
- 序列化对象
- 打印流
- **补充知识：Properties**
- 补充知识：IO框架

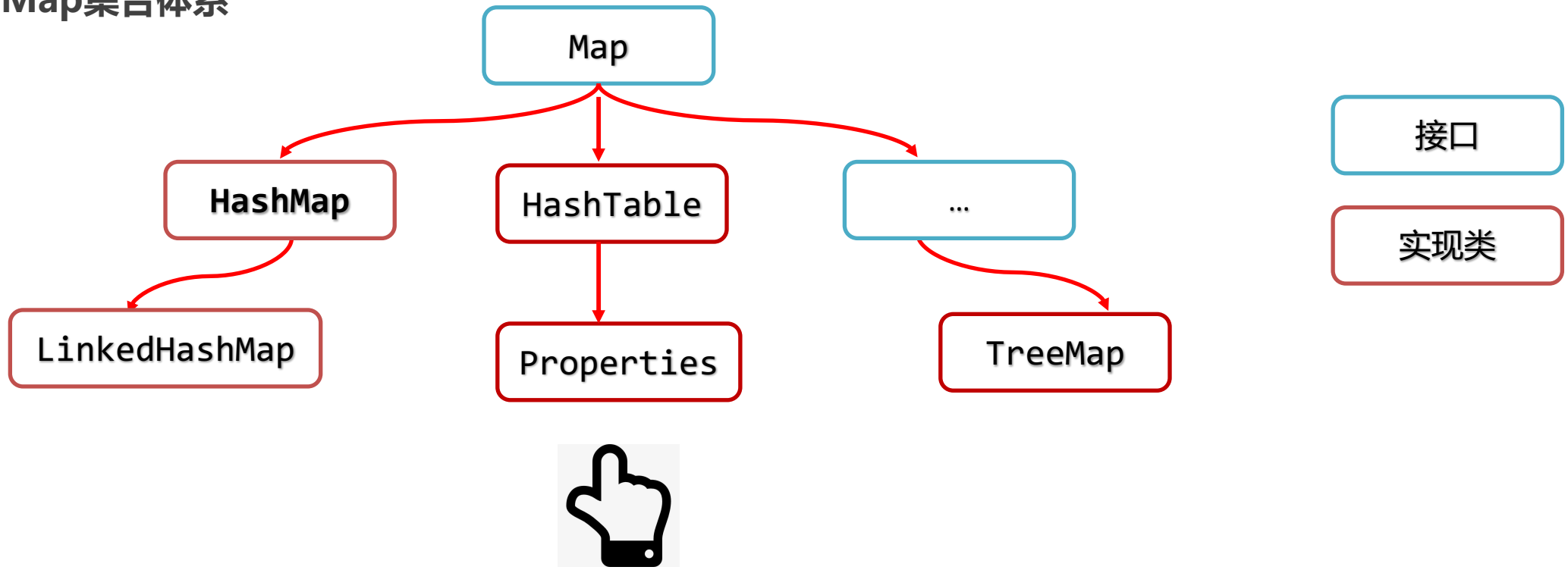
以后我们系统中，可能会有一个.properties结尾的属性文件



怎么用程序生成这个属性文件？

怎么读取这个属性文件里面的内容？

Map集合体系


Map
HashMap
HashTable
…
LinkedHashMap
Properties
TreeMap
接口
实现类

Properties属性集对象

- 其实就是一个Map集合，但是我们一般不会当集合使用，因为HashMap更好用。

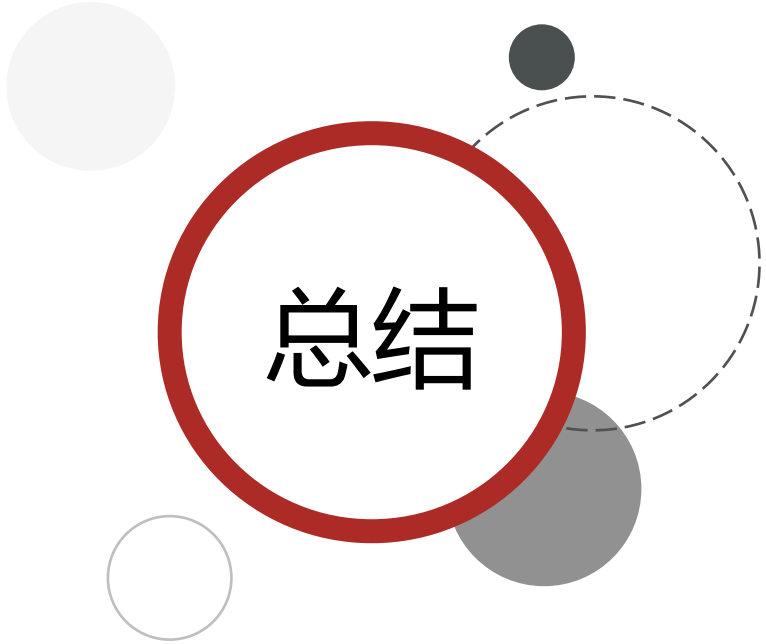
Properties核心作用：

- Properties代表的是一个属性文件，可以把自己对象中的键值对信息存入到一个属性文件中去。
- 属性文件：后缀是.properties结尾的文件,里面的内容都是 key=value，后续做系统配置信息的。

Properties的API:

- Properties和IO流结合的方法：

构造器	说明
void load (InputStream inStream)	从输入字节流读取属性列表（键和元素对）
void load (Reader reader)	从输入字符流读取属性列表（键和元素对）
void store (OutputStream out, String comments)	将此属性列表（键和元素对）写入此 Properties表中，以适合于使用 load(InputStream)方法的格式写入输出字节流
void store (Writer writer, String comments)	将此属性列表（键和元素对）写入此 Properties表中，以适合使用 load(Reader)方法的格式写入输出字符流
public Object setProperty(String key, String value)	保存键值对（put）
public String getProperty(String key)	使用此属性列表中指定的键搜索属性值 (get)
public Set<String> stringPropertyNames()	所有键的名称的集合 (keySet())



1. Properties的作用？

- **可以存储Properties属性集的键值对数据到属性文件中去：**
  - **void store(Writer writer, String comments)**
- **可以加载属性文件中的数据到Properties对象中来：**
  - **void load(Reader reader)**

练习

拷贝文件夹

需求：将某个磁盘的文件夹拷贝到另一个文件夹下去，包括文件夹中的全部信息

分析：

①：IO默认不可以拷贝文件夹

②：我们需要遍历文件夹，如果是文件则拷贝过去，如果是文件夹则要进行1-1创建，再递归。

练习

删除文件夹

需求：将某个磁盘的文件夹拷贝到另一个文件夹下去，包括文件夹中的全部信息

分析：

①：IO默认不可以拷贝文件夹

②：我们需要遍历文件夹，如果是文件则拷贝过去，如果是文件夹则要进行1-1创建，再递归。

练习

点名器

需求：有一个文件里面存储了班级同学的姓名，每一个姓名占一行，要求通过程序实现随机点名器。

思路：

① 把文件中的数据读取到到集合中

② 使用Random产生一个随机数，获得随机索引。

③ 通过随机索引获取随机姓名

练习

点名器升级版

需求：有一个文件里面存储了班级同学的姓名，每一个姓名占一行，要求通过程序实现随机点名器。

第三次必定是张三同学

思路：

① 第一次运行时随机的姓名

② 第二次运行时随机的姓名

③ 第三次运行时随机的姓名

难点：

如何确定当前是第几次运行程序？

练习

登录案例

需求：写一个登陆小案例。

步骤：

① 将正确的用户名和密码手动保存在本地的userinfo.txt文件中。
② 保存格式为:username=zhangsan&password=123
③ 让用户键盘录入用户名和密码
④ 比较用户录入的和正确的用户名密码是否一致
⑤ 如果一致则打印登陆成功
⑥ 如果不一致则打印登陆失败

练习

自动登录案例

需求：写一个自动登陆小案例。

步骤：

① 将正确的用户名和密码手动保存在本地的userinfo.txt文件中。

② 保存格式为:username=zhangsan&password=123

③ 让用户键盘录入用户名和密码

④ 比较用户录入的和正确的用户名密码是否一致

⑤ 如果一致则打印登陆成功，并将用户录入的数据保存到本地cookie.txt文件中。

保存格式为：username=zhangsan

password=123

⑤ 如果不一致则打印登陆失败

⑥ 再次运行时，则从本地cookie.txt文件中读取第一次保存的数据，实现自动登陆。

目录

Contents

- 缓冲流
- 转换流
- 序列化对象
- 打印流
- 补充知识：Properties
- 补充知识：IO框架

commons-io概述

- commons-io是apache开源基金组织提供的一组有关IO操作的类库，可以提高IO功能开发的效率。
- commons-io工具包提供了很多有关io操作的类。有两个主要的类FileUtils, IOUtils

FileUtils主要有如下方法:

方法名	说明
`String readFileToString(File file, String encoding)`	读取文件中的数据, 返回字符串
`void copyFile(File srcFile, File destFile)`	复制文件。
`void copyDirectoryToDirectory(File srcDir, File destDir)`	复制文件夹。

步骤

导入commons-io-2.6.jar做开发

需求

- 使用commons-io简化io流读写

分析

① 在项目中创建一个文件夹：lib

② 将commons-io-2.6.jar文件复制到lib文件夹

③ 在jar文件上点右键，选择 Add as Library -> 点击OK

④ 在类中导包使用

传智教育旗下高端IT教育品牌