Java I/O在Android中应用

T H A N K Y O U F O R W A T C H I N G

主讲老师Alvin：2464061231

为什么要学习Java I/O

- 扎实的Java基础：

  在对象序列化/Json解析/XML解析

  zip压缩均需要以I/O作为其基础

  处理底层数据业务的时候

讲师简介

Alvin

华南理工大学 软件工程 工程硕士

三星中国研究院 5 years
项目经理

小米科技 2 years
技术总监

- 曾就业于三星中国研究院及小米旗下互联网公司担任android任软件工程师及项目经理
- 拥有扎实的C/Java 基础，深入研究android系统多年。
- 讲课形象生动，热情洋溢

目 录

CONTENTS

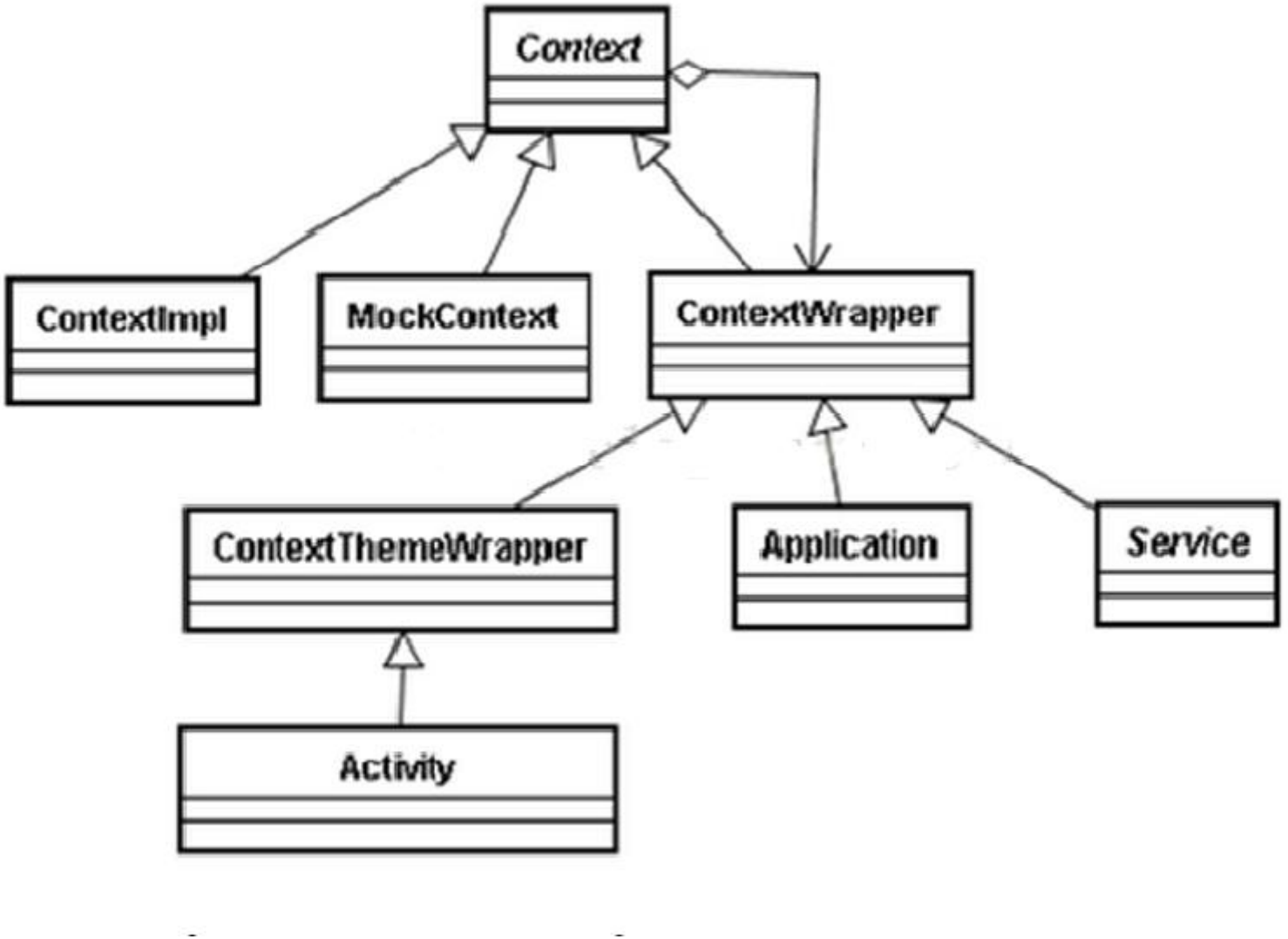
01 IO中经常看见的使用方式

```
DataOutputStream out = new DataOutputStream(
                new BufferedOutputStream(
                new FileOutputStream(
                new File(file))));
```

这种嵌套是不是经常绕晕了？

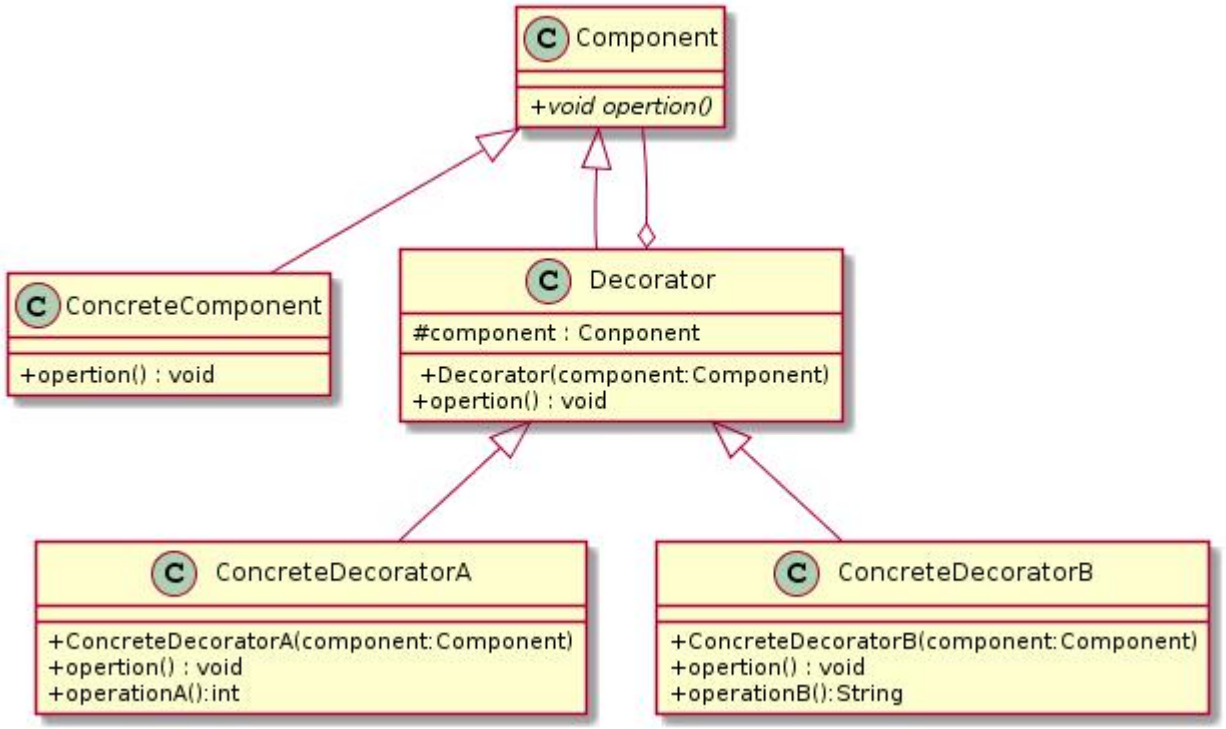
这个嵌套的原理是什么？

01 装饰模式之Android源码



01 装饰模式

01 装饰模式



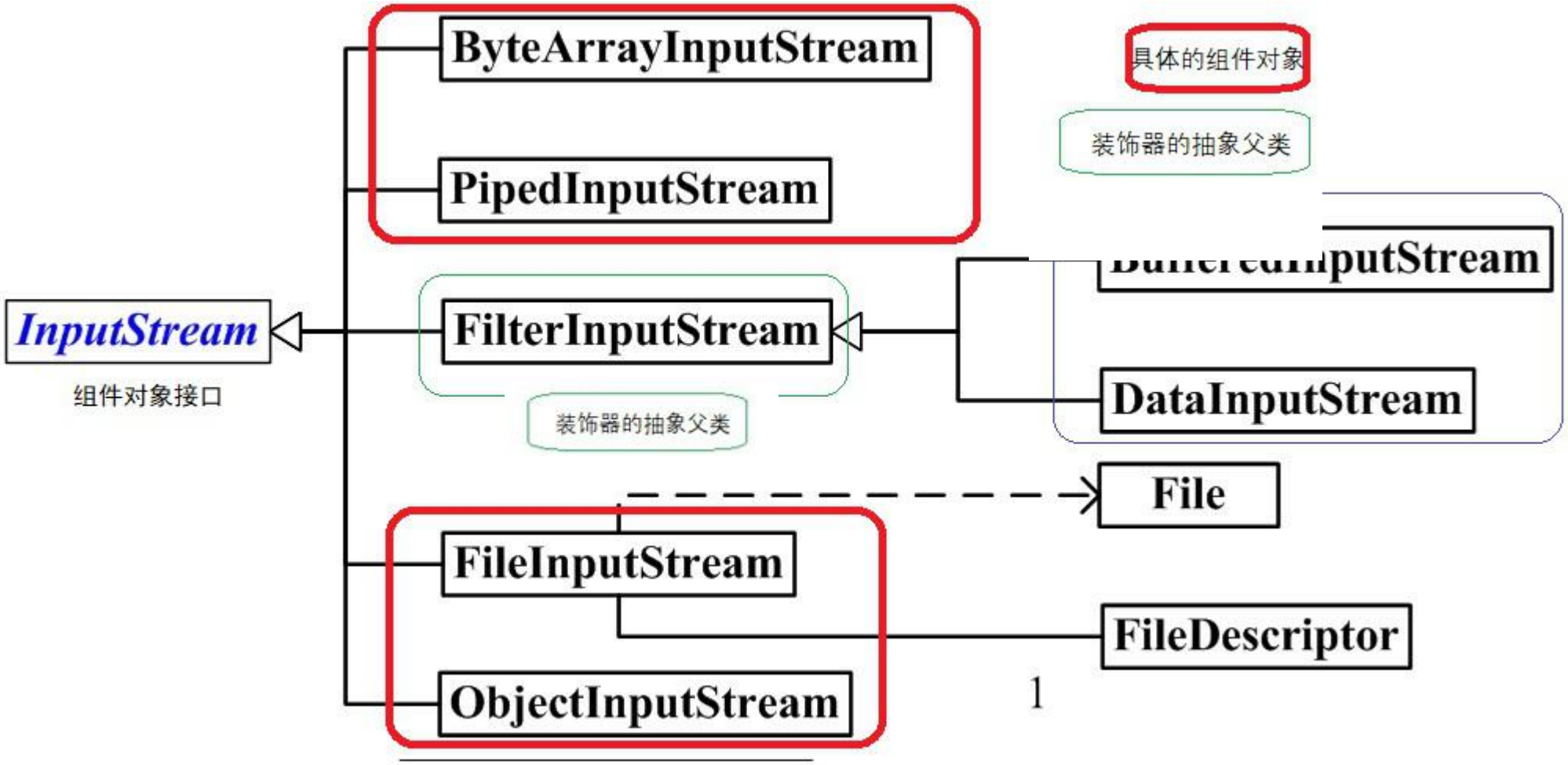
Component：抽象构建接口

ConcreteComponent：具体的构建对象，实现组件对象接口，通常就是被装饰的原始对象。就对这个对象添加功能。

Decorator：所有装饰器的抽象父类，需要定义一个与组件接口一致的接口，内部持有一个Component对象，就是持有一个被装饰的对象。

ConreteDecoratorA/ConreteDecoratorB：实际的装饰器对象，实现具体添加功能。熟悉UML图的基本就明白了，但是像我这样不熟悉的还是写一点我们熟悉的代码描述吧。

01 IO中的装饰器模式



01 IO学习的关键方法

挑战:Java IO的学习是一件非常艰巨的任务。

全面：它的挑战是来自于要覆盖所有的可能性。不仅存在各种I/O源端还有想要和他通信的接收端（文件/控制台/网络链接），而且还需要以不同的方式与他们进行通信（顺序/随机存取/缓冲/二进制/字符/行/字 等等）这些情况综合起来就给我们带来了大量的学习任务，大量的类需要学习。

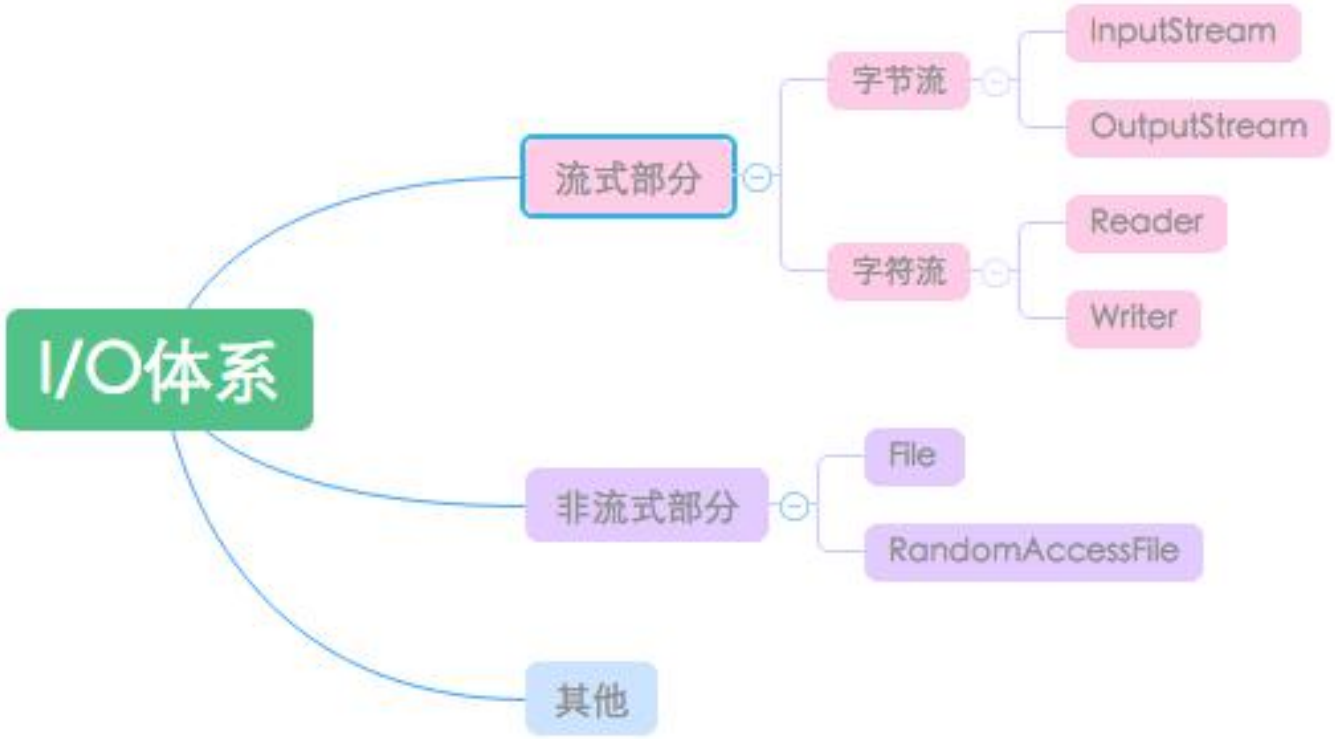
历史：我们要学会所有的这些java 的IO是很难的，因为我们没有构建一个关于IO的体系，要构建这个体系又需要深入理解IO库的演进过程，所以，我们如果缺乏历史的眼光，很快我们会对什么时候应该使用IO中的哪些类，以及什么时候不该使用它们而困惑。

所以，在开发者的眼中，IO很乱，很多类，很多方法，很迷茫。

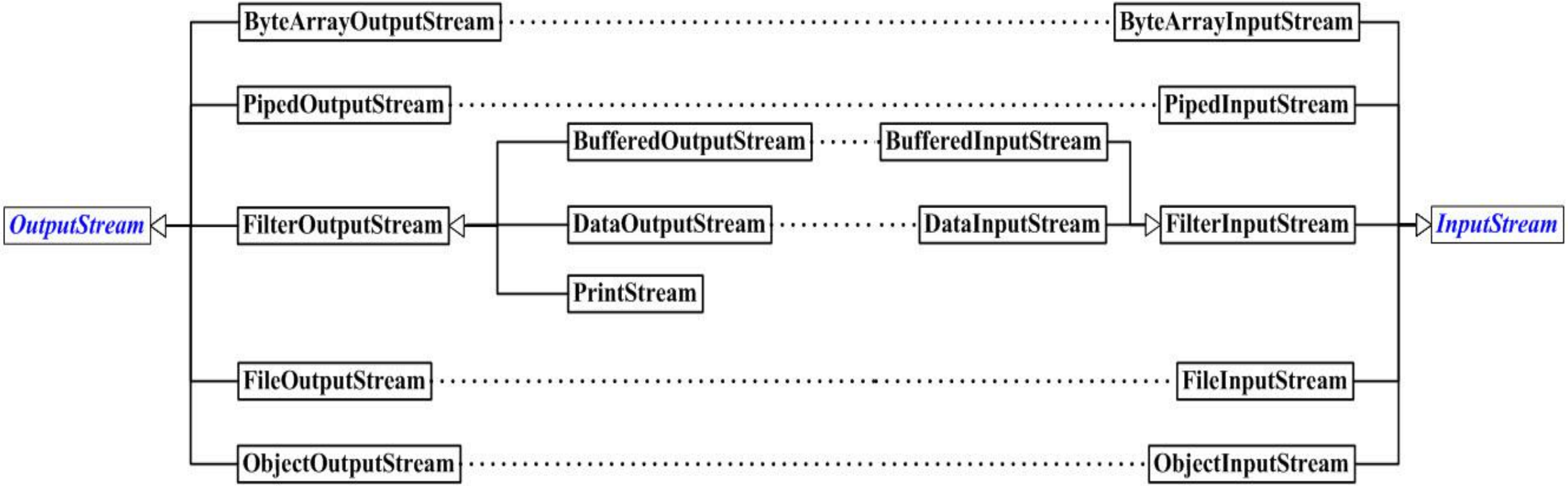
目 录

CONTENTS

01 IO类结构

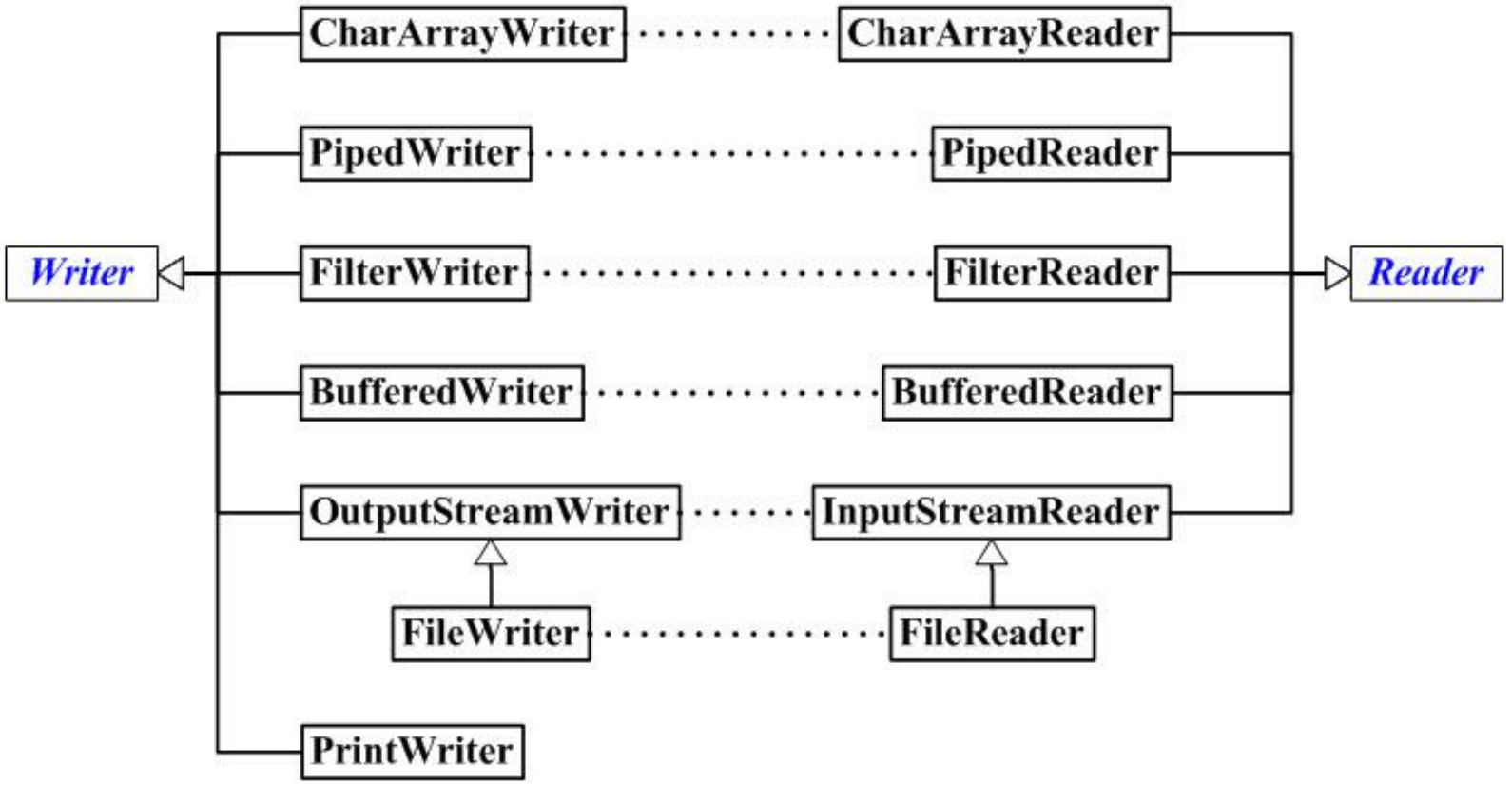

I/O体系
流式部分
字节流
InputStream
OutputStream
字符流
Reader
Writer
非流式部分
File
RandomAccessFile
其他

01 字节流的学习



OutputStream -> FileOutputStream/FilterOutputStream ->DataOutputStream->bufferedOutputStream

01 字符流的学习



Writer- >FilterWriter->BufferedWriter->OutputStreamWriter->FileWriter->其他

字符有一个 readline() ，且对于中文的区别

01 字符流

```
BufferedWriter bufferedWriter = new BufferedWriter(
        new OutputStreamWriter(
                new FileOutputStream(
                        new File("src/testtxt/writerAndStream.txt")),"GBK"));

bufferedWriter.write("我爱你中国，亲爱的母亲");
bufferedWriter.flush();
bufferedWriter.close();
```

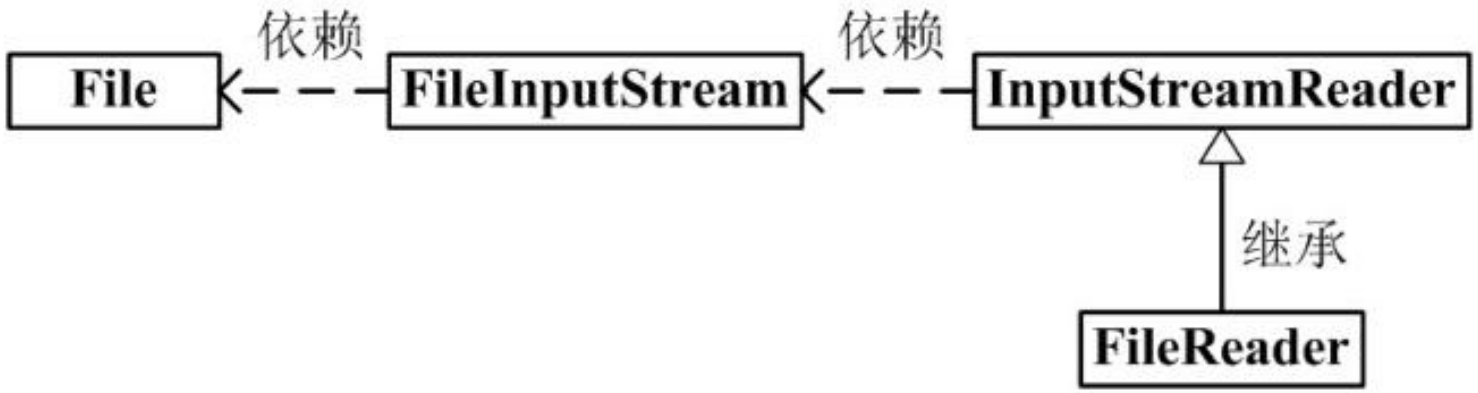
1：如果只用FileOutputStream fileOutputStream = new FileOutputStream("d:/text.txt");不是也能输出到"d:/text.txt"吗？为什么要用其它两个呢？能起到什么作用呢？
答案：
FileOutputStream　　：是字节流，它一个字节一个字节的向外边送数据
OutputStreamWriter：是字符流，它一个字符一个字符的向外边送数据

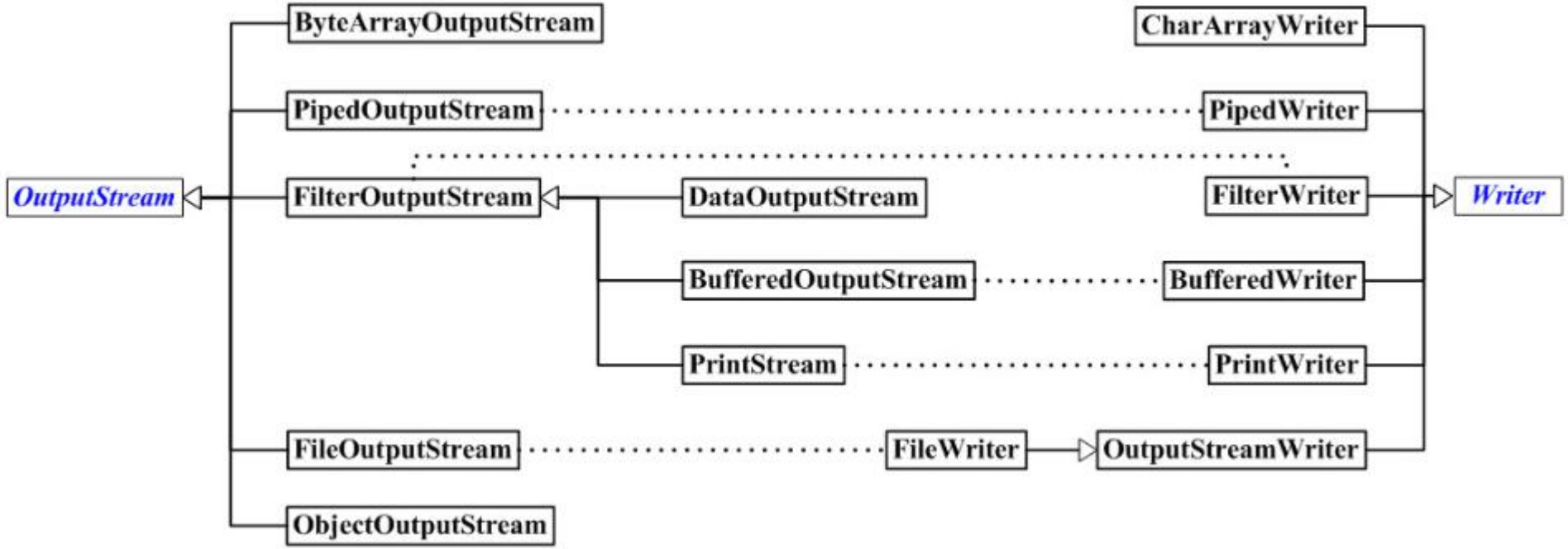
2：它们有什么区别么？
答案：
因为计算机是洋鬼子发明的，它们的英文字符占一个字节，而我们的中文是一个字符，至少占俩字节。如果用stream，你读出来的英语再倒也罢了，读出来的中文可就是乱码或者一个个“？？？？”。
如果你用WRITER，就不会有乱码了

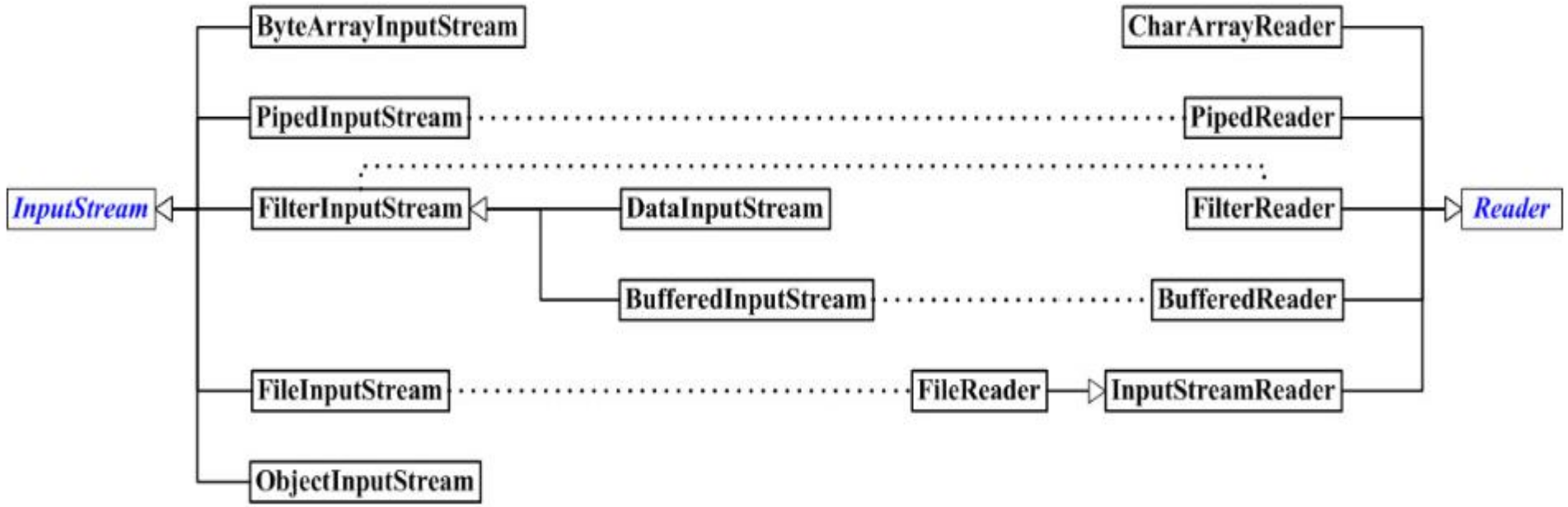
3：BufferedWriter　Buffer是一个缓冲区，为什么要用BUFFER呢？
答案：
如果你直接用stream或者writer，你的硬盘可能就是读一个字符或者一个字节　　就去读写硬盘一次，IO负担巨大。可是你用了Buffer，你的硬盘就是读了一堆数据之后，读写一下硬盘。这样对你硬盘有好处。

01 字符流最常见用法

在java中，字节流能转换为字符流，下面是它们的转换关系图。




ByteArrayOutputStream
CharArrayWriter
PipedOutputStream
PipedWriter
OutputStream
FilterOutputStream
DataOutputStream
FilterWriter
Writer
BufferedOutputStream
BufferedWriter
PrintStream
PrintWriter
FileOutputStream
FileWriter
OutputStreamWriter
ObjectOutputStream


ByteArrayInputStream
CharArrayReader
PipedInputStream
PipedReader
InputStream
FilterInputStream
DataInputStream
FilterReader
Reader
BufferedInputStream
BufferedReader
FileInputStream
FileReader
InputStreamReader
ObjectInputStream

目 录

CONTENTS

01 RandomAccessFile

为什么要有RandomAccessFile?

多线程中分段下载

FileWriter/
FileReader
?

常用方法简介?

构造方法：RandomAccessFile raf = newRandomAccessFile(File file, String mode);
其中参数 mode 的值可选 "r"：可读，"w" ：可写，"rw"：可读性;

成员方法：

seek(int index);可以将指针移动到某个位置开始读写;
setLength(long len);给写入文件预留空间：

01 RandomAccessFile 特点和优势

1.既可以读也可以写

RandomAccessFile不属于InputStream和OutputStream类系的它是一个完全独立的类，所有方法(绝大多数都只属于它自己)都是自己从头开始规定的,这里面包含读写两种操作

2.可以指定位置读写

RandomAccessFile能在文件里面前后移动，在文件里移动用的seek( ),所以它的行为与其它的I/O类有些根本性的不同。总而言之，它是一个直接继承Object的，独立的类。只有RandomAccessFile才有seek搜寻方法，而这个方法也只适用于文件.

01 NIO——FileChannel

Channel是对I/O操作的封装。

FileChannel配合着ByteBuffer，将读写的数据缓存到内存中，然后以批量/缓存的方式read/write，省去了非批量操作时的重复中间操作，操纵大文件时可以显著提高效率（和Stream以byte数组方式有什么区别？经过测试，效率上几乎无区别）。

01 NIO——FileChannel

目 录

CONTENTS

01 作业

1：建议：学习 Java编程思想 中的IO章节部分

2：系统的总结流的用法以及类的结构图 写代码

3：将APP加固代码进行预先