# 第二章总结

## 文件读写缓冲（字节流）

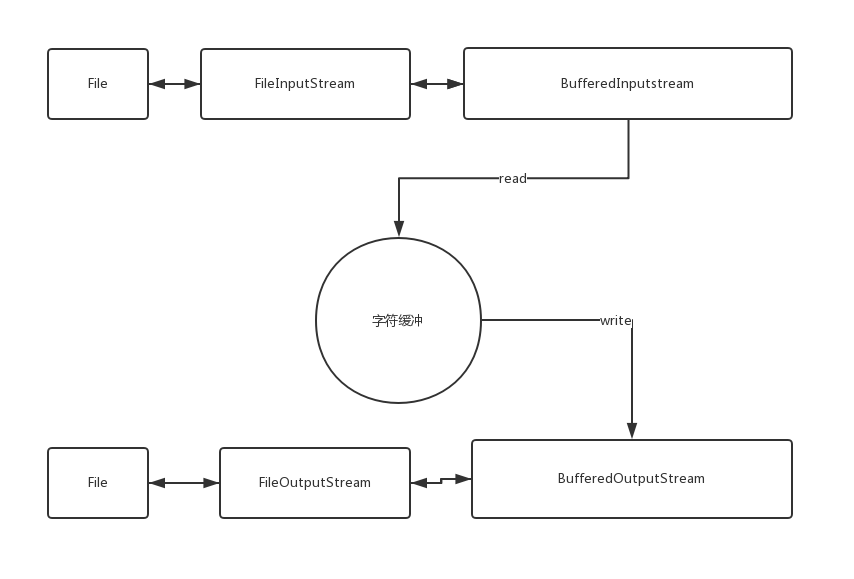
tip：输入和输出的参照物是你当前的正在写的程序

将文件路径传入可操作文件对象======》可操作文件对象传入输出字节流======》字节流转

换为输出缓冲流======》将文件内容读到输出缓冲返回一个字符整数

将文件路局传入可操作文件对象======》可操作文件对象传入输入字节流======》字节流转

换为输入缓冲流======》将文件内容写到输出缓冲返回一个字符整数



## 2. 序列化与反序列化

实现序列化首先要进行序列化对象所在的类实现：



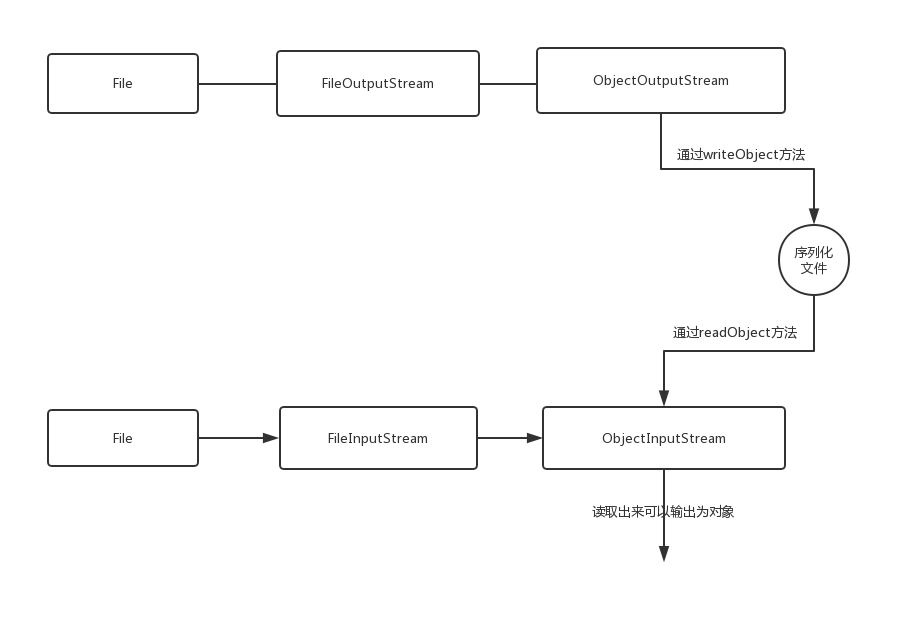
### 2.1序列化

#### 2.1.1简述

是指将对象以流的形式存储（输出）在文件中

#### 2.1.2具体过程：

新建一个可操作文件对象==》将可操作文件对象传入 文件输出流==》文件输出流转化为序列化输出流==》然后调用序列化输出流对象的方法将对象写入文件

图示：

### 2.2反序列化

#### 2.2.1简述

是指将已经序列化了的存储了对象的文件转化为（输入）对象

1. **public** **class** Demo1  {
2. **public** **static** **void** main(String[] args) **throws** IOException, ClassNotFoundException {
3. Student student1 = **new** Student(101, "小一", "男", "上海");
4. Student student2 = **new** Student(102, "小二", "女", "香港");
5. Student student3 = **new** Student(103, "小三", "男", "深圳");
6. Student student4 = **new** Student(104, "小四", "女", "东莞");
7. //定义 将对象将写入的 文件输出流对象
8. FileOutputStream fileOutputStream = **new** FileOutputStream(**new** File("c:/student.txt"));
9. //定义 对象序列化输出流
10. ObjectOutputStream objectOutputStream = **new** ObjectOutputStream(fileOutputStream);
11. //将对象序列化
12. objectOutputStream.writeObject(student1);
13. objectOutputStream.writeObject(student2);
14. objectOutputStream.writeObject(student3);
15. objectOutputStream.writeObject(student4);
16. //关闭流
17. objectOutputStream.close();
18. System.out.println("序列化成功");
20. //将对象反序列化
21. //定义一个文件输入流对象
22. FileInputStream fileInputStream = **new** FileInputStream(**new** File("c:/student.txt"));
23. //定义一个对象输入流对象
24. ObjectInputStream objectInputStream = **new** ObjectInputStream(fileInputStream);
25. //将对象流反序列化
26. **for** (**int** i = 0; i < 4; i++) {
27. Student student=(Student) objectInputStream.readObject();
28. System.out.println(student);
29. }
30. System.out.println("反序列化成功");
31. //关闭流
32. objectInputStream.close();
33. objectOutputStream.close();
35. }
36. }

## 3.字符缓冲流

BufferedReader 与PrintWriter