Java Stream、File、IO

Java Stream

> Java Stream 是 Java 8 中引入的一种新的抽象数据类型，它允许开发人员使用函数式编程的方式来处理集合数据。
>
> 使用 Java Stream 可以方便地进行过滤、映射、排序和聚合等操作。下面是一个简单的示例：
>
> 该示例将一个整数列表中的偶数筛选出来，并将它们加起来。

Java | 复制代码

```java
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5);

int sum = numbers.stream()
                 .filter(n -> n % 2 == 0)
                 .mapToInt(Integer::intValue)
                 .sum();

System.out.println("The sum of even numbers: " + sum);

```

Java File

> Java File 类是用于表示文件或目录的抽象路径名的类。它可以用于创建、删除、重命名和移动文件或目录。

以下是一些常用的 File 方法：

- **exists()**：检查文件或目录是否存在
- **isFile()**：检查文件是否存在且是一个文件
- **isDirectory()**：检查文件是否存在且是一个目录
- **createNewFile()**：创建一个新文件
- **delete()**：删除文件或目录

- **renameTo(File dest)**：将文件或目录重命名为给定的目标路径名

> 以下是一个示例，演示如何使用 File 类创建一个新文件：
>
> 该示例创建一个名为 "example.txt" 的新文件。如果该文件已经存在，则输出 "File already exists."。

Java | 复制代码

```java
File file = new File("example.txt");
try {
    if (file.createNewFile()) {
        System.out.println("File created: " + file.getName());
    } else {
        System.out.println("File already exists.");
    }
} catch (IOException e) {
    System.out.println("An error occurred.");
    e.printStackTrace();
}
```

Java IO

> Java IO（Input/Output）是 Java 中用于读写数据的标准输入输出库。它包括字节流和字符流两种类型，分别用于读写二进制数据和文本数据。

以下是一些常用的 IO 类：

- **InputStream**：字节输入流
- **OutputStream**：字节输出流
- **Reader**：字符输入流
- **Writer**：字符输出流

> 以下是一个示例，演示如何使用 IO 类从文件中读取数据并将其打印到控制台：
>
> 该示例打开一个名为 "example.txt" 的文件，并从中读取数据。使用 BufferedReader 可以方便地按行读取数据，并将其打印到控制台。

Java | 复制代码

```java
try (FileInputStream fis = new FileInputStream("example.txt");
     InputStreamReader isr = new InputStreamReader(fis);
     BufferedReader br = new BufferedReader(isr)) {
    String line;
    while ((line = br.readLine()) != null) {
        System.out.println(line);
    }
} catch (IOException e) {
    System.out.println("An error occurred.");
    e.printStackTrace();
}
```