📖 青空的B站课程

📅 2024年4月20日

字符流

⭐ 字节流和字符流不一样，它是具体的一个字符一个字符来进行操作，它只适合读取纯文本的文件，不适合读取媒体文件，使用`FileReader`

```
    public static void main(String[] args) {
        try(FileReader fileReader = new FileReader("test.txt")){
            fileReader.skip(2);
            System.out.println((char) fileReader.read()); //和字节流差不多
        }catch (IOException e){
            throw  new RuntimeException();
        }
    }
```

⭐ 但是字符流如果要读取多个的字符就需要使用到`char[]`

```
    public static void main(String[] args) {
        try(FileReader fileReader = new FileReader("test.txt")){
            char[] chars = new char[10]; //创建char数组
            fileReader.read(chars); //读取
            System.out.println(chars); //输出chars字符
         }catch (IOException e){
            throw  new RuntimeException();
        }
    }
```

⭐ 有输入就会有输出输出使用`FileWrite`，这里是演示了一个输入一个`char`数组

```
    public static void main(String[] args) {
        try(FileWriter fileWriter = new FileWriter("test.txt",true)){ //可以追加
            fileWriter.write("HelloWorld!".toCharArray()); //转换成Char数组
         }catch (IOException e){
            throw  new RuntimeException();
        }
    }
```

1️⃣ 它还有个`append`，它和`write`差不多，但是`append`支持像`StringBuilder`那样的链式调用，返回的是`Writer`对象本身

```
    fileWriter.append("你好").append("我是").append("lbw");
```

主要用法查询

当然，以下是一个关于Java中`FileReader`和`FileWriter`类（字符流）的方法及使用说明的表格：

类名	方法名称	描述
`FileReader`		
构造方法	`FileReader(File file)`	创建一个`FileReader`对象，用于从指定File对象表示的文件读取字符
	`FileReader(String fileName)`	创建一个`FileReader`对象，用于从指定路径的文件读取字符
读取方法	`int read()`	读取一个字符，返回读取到的字符（如果是EOF则返回-1）
	`int read(char[] cbuf)`	读取字符到字符数组`cbuf`中，返回实际读取的字符数
	`int read(char[] cbuf, int off, int len)`	从文件读取`len`个字符到`cbuf`数组的off偏移量处开始的位置
关闭方法	`void close()`	关闭此文件读取流并释放与之关联的所有系统资源

类名	方法名称	描述
`FileWriter`		
构造方法	`FileWriter(File file)`	创建一个`FileWriter`对象，用于向指定File对象表示的文件写入字符
	`FileWriter(String fileName)`	创建一个`FileWriter`对象，用于向指定路径的文件写入字符
写入方法	`void write(int c)`	将指定的字符写入此输出流
	`void write(char[] cbuf)`	将指定字符数组中的所有字符写入此输出流
	`void write(char[] cbuf, int off, int len)`	将指定字符数组的一部分（从off偏移量开始，长度为`len`）写入此输出流
	`void write(String str)`	将指定字符串的所有字符写入此输出流
	`void write(String str, int off, int len)`	将指定字符串的一部分（从off偏移量开始，长度为`len`）写入此输出流
关闭方法	`void close()`	关闭此文件输出流并释放与此流相关联的所有系统资源

此外，这里有一个使用`FileReader`和`FileWriter`实现文件内容复制的简单示例：

```java
public void copyFileUsingCharacterStream(String source, String destination)
throws IOException {
    FileReader reader = new FileReader(source);
    FileWriter writer = new FileWriter(destination);

    int c;
    while ((c = reader.read()) != -1) {
        writer.write(c);
    }

    reader.close();
    writer.close();
}
```

在这个例子中，首先创建`FileReader`和`FileWriter`对象分别指向源文件和目标文件，然后逐字符读取并写入到另一个字符流中，直到读取完毕。最后，关闭两个流以释放资源。注意，字符流默认使用平台的默认字符编码进行读写操作。如果需要指定特定的字符编码，应使用`InputStreamReader`和`OutputStreamWriter`配合`FileInputStream`和`FileOutputStream`使用。