Dec 13, 2021

# How to read and write binary files in Java?

## What is a binary file?

A **binary file** is a computer file that is not a text file. The term "binary file" is often used as a term meaning "non-text file". Many binary file formats contain parts that can be interpreted as text

## Methods for reading a binary file:

**Reading and Writing Binary Files Using FileInputStream and FileOutputStream:**

The following examples use the **FileInputStream** and **FileOutputStream** classes to perform low level binary I/O.

Example:

```
import java.io.*;

class Main {
    public static void main (String[] args) throws IOException {

        String fileName="out.bin";
        FileOutputStream fileOutputStream=new
FileOutputStream(fileName);
        ObjectOutputStream objectOutputStream=new
ObjectOutputStream(fileOutputStream);
        objectOutputStream.writeInt(2000);
        objectOutputStream.writeInt(3000);
        objectOutputStream.writeInt(4000);
        objectOutputStream.close();
        System.out.println("Done Writing");
```

```
        FileInputStream fileInputStream=new FileInputStream(fileName);
        ObjectInputStream objectInputStream=new
ObjectInputStream(fileInputStream);
        System.out.println(objectInputStream.readInt());
        System.out.println(objectInputStream.readInt());
        System.out.println(objectInputStream.readInt());
        System.out.println("Done Reading");
        objectInputStream.close();

    }
}
```

Output:

```
Done Writing
2000
3000
4000
Done Reading
```

**Reading and Writing Binary Files Using BufferedInputStream and BufferedOutputStream:**

The following examples use the **BufferedInputStream** and **BufferedOutputStream** classes to perform low level binary I/O:

Using **BufferedInputStream** and **BufferedOutputStream** is as same as **FileInputStream** and **FileOutputStream**. The only difference is that a buffered stream uses an array of byte internally to buffer the input nd output to reduce the number of calls to the native API, hence increasing IO performance.

By default, both **BufferedInputStream** and **BufferedOutputStream** has an internal buffer of 8192 bytes (8KB), but we can specify a custom buffer size at initialization

Example:

```
import java.io.*;

class Main {
    public static void main (String[] args) throws IOException {

        String fileName="out.bin";
        BufferedOutputStream bufferedOutputStream=new
BufferedOutputStream(new FileOutputStream(fileName));
        bufferedOutputStream.write(2000);
        bufferedOutputStream.write(3000);
        bufferedOutputStream.write(4000);
        bufferedOutputStream.close();
        System.out.println("Done Writing");
        BufferedInputStream bufferedInputStream=new
BufferedInputStream(new FileInputStream(fileName));
        System.out.println(bufferedInputStream.read());
        System.out.println(bufferedInputStream.read());
        System.out.println(bufferedInputStream.read());
        System.out.println("Done Reading");
        bufferedOutputStream.close();

    }
}
```

Output:

```
Done Writing

208

184
```

```
160

Done Reading
```

## How does CodersArts help you in Java coding?

### CodersArts provide :

- Java assignment Help
- Help in Java development Projects
- Mentorship from Experts Live 1:1 session
- Course and Project completions
- CourseWork help

Contact Us