Advanced Programming

File, Stream & Serialization in Java

Instructor: Ali Najimi

Author: Hossein Masihi

Department of Computer Engineering

Sharif University of Technology

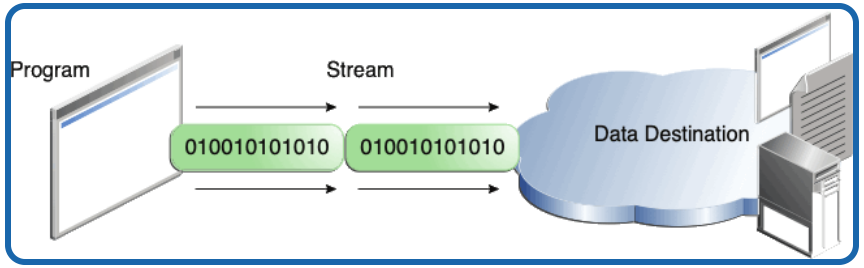
Fall 2025

Table of Contents

Streams — Concept

- A **stream** is a flow of data.
- Java I/O works through **streams**:
  - InputStream → read data
  - OutputStream → write data
- Streams are **sequential** data channels.

> Everything in Java I/O is built around streams.




File IO with Java 8 Streams
nextptr

Input & Output Overview

Type	Parent Class	Purpose
Byte Stream	`InputStream`, `OutputStream`	Handles raw binary data
Character Stream	`Reader`, `Writer`	Handles text characters with encoding

```
InputStream in = new FileInputStream("data.bin");
Reader reader = new FileReader("data.txt");
```

Reading a File (Character Stream)

```
try (BufferedReader br = new BufferedReader(new FileReader("input.txt"))) {
    String line;
    while ((line = br.readLine()) != null) {
        System.out.println(line);
    }
}
```

- `BufferedReader` improves efficiency
- `try-with-resources` auto-closes the file

Writing to a File

```
try (BufferedWriter bw = new BufferedWriter(new FileWriter("output.txt"))) {
    bw.write("Sharif University");
}
```

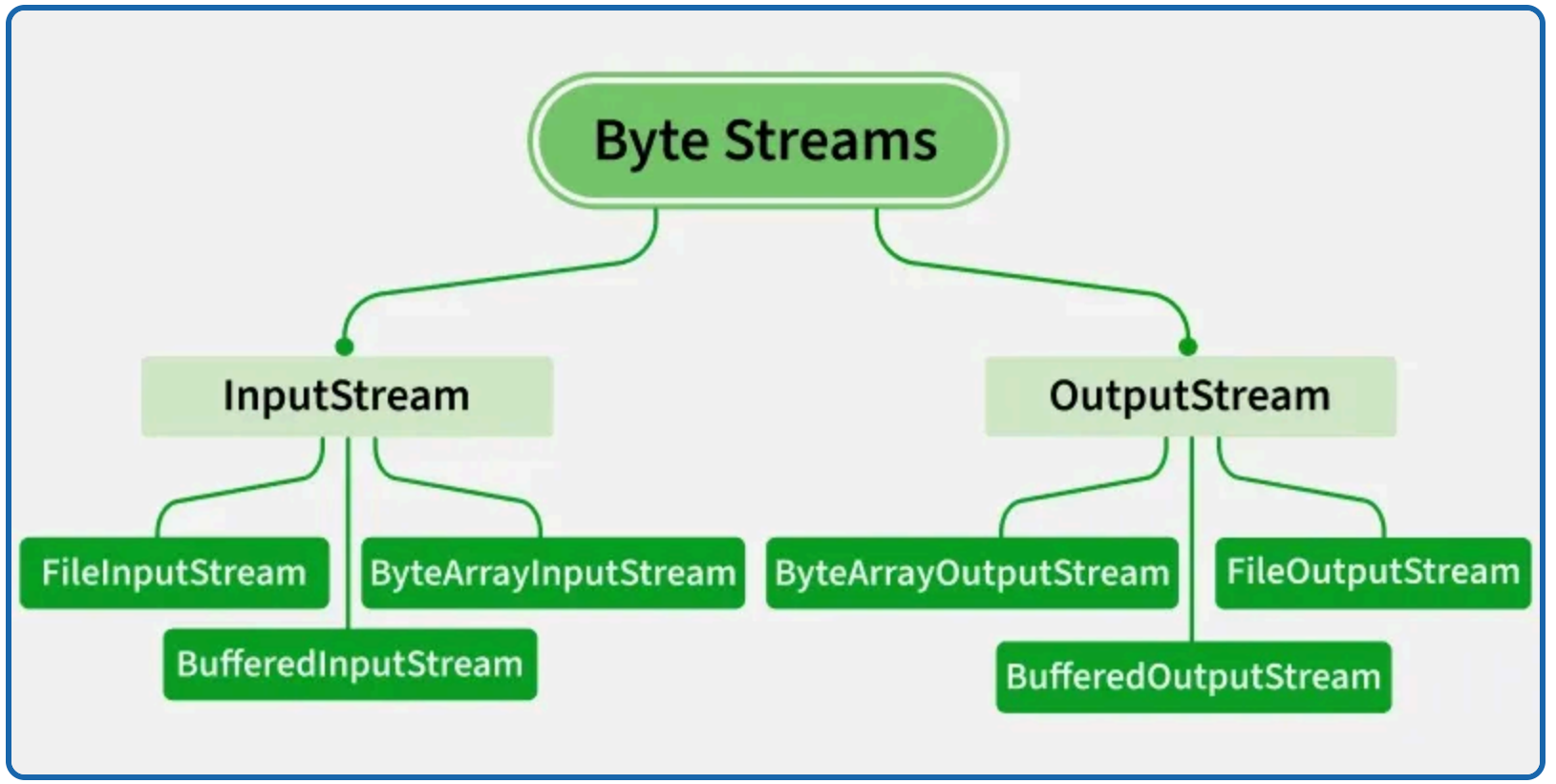
- Writing happens in **text form**
- Buffered writers reduce disk operations

Byte Streams — Handling Raw Data

```
try (FileInputStream in = new FileInputStream("image.png");
     FileOutputStream out = new FileOutputStream("copy.png")) {

    int data;
    while ((data = in.read()) != -1) {
        out.write(data);
    }
}
```

> Used for images, audio, compiled files, etc.


Byte Streams
InputStream
OutputStream
FileInputStream
ByteArrayInputStream
BufferedInputStream
ByteArrayOutputStream
FileOutputStream
BufferedOutputStream

Serialization — Storing Objects

- Serialization converts an **object into a byte stream**
- Allows:
  - Saving objects to disk
  - Sending objects over network

```
class Student implements Serializable {
    String name;
    int id;
}
```

Writing Serialized Objects

```
ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream("std.bin"));
out.writeObject(new Student("Ali", 401110891));
out.close();
```

Reading Serialized Objects

```
ObjectInputStream in = new ObjectInputStream(new FileInputStream("std.bin"));
Student s = (Student) in.readObject();
in.close();
```

> Requires class to implement `Serializable`.

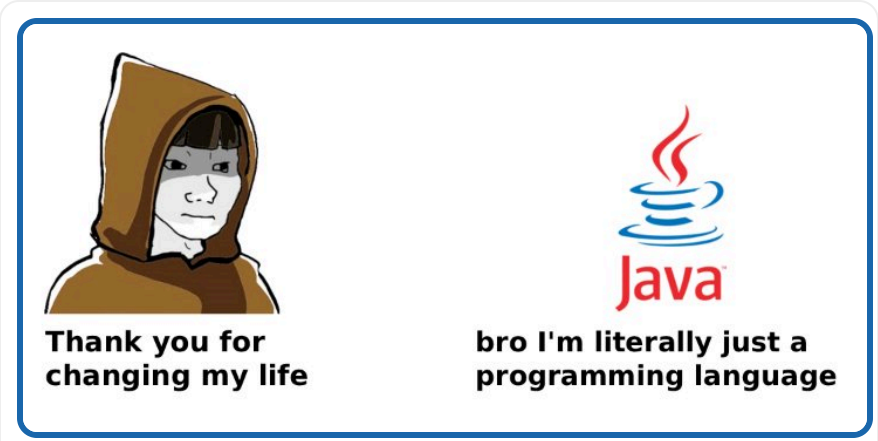
Summary

Concept	Description
Stream	Sequential data flow
Byte Stream	Raw data (binary)
Character Stream	Encoded text
File I/O	Reading and writing text and binary files
Serialization	Converting objects → transferable byte streams

> Mastering I/O is essential for data processing applications.

Thank You!

Java File & Stream — Practical I/O



Sharif University of Technology — Advanced Programming — Fall 2025