Input and output stream in java-

Java I/O (Input and Output) is used *to* process the input *and* produce the output.

Java uses the concept of a stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations.

We can perform **file handling in Java** by Java I/O API.

Stream

A stream is a sequence of data. In Java, a stream is composed of bytes.

In Java, 3 streams are created for us automatically. All these streams are attached with the console.

1) System.out: standard output stream

2) System.in: standard input stream

3) System.err: standard error stream

Let's see the code to print **output and an error** message to the console.

```
1. System.out.println("simple message");
2. System.err.println("error message");
```

The explanation of OutputStream and InputStream classes are given below:

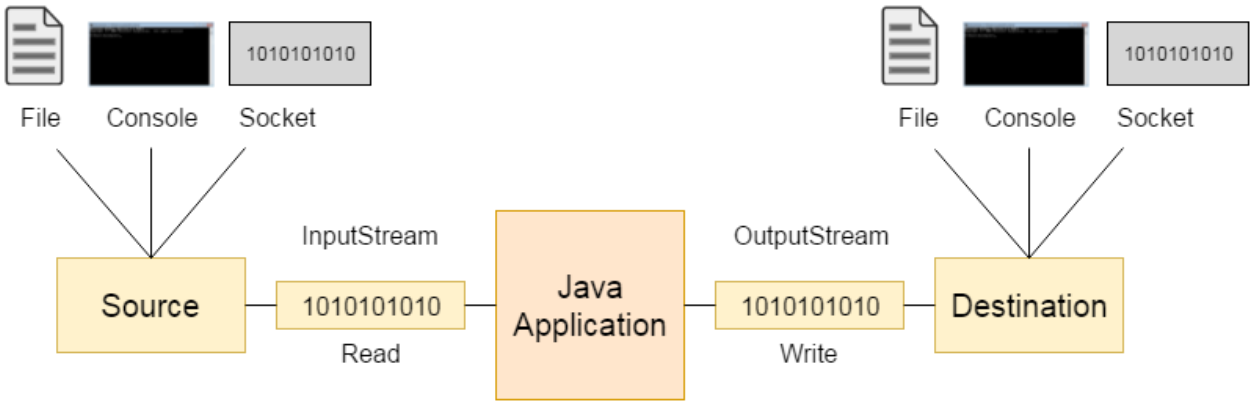
OutputStream

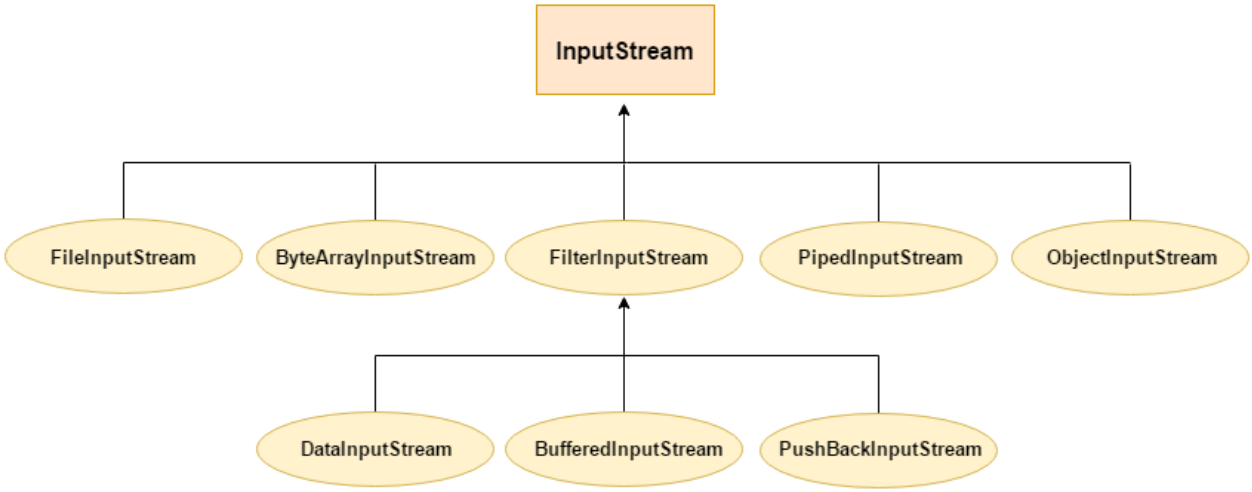
Java application uses an output stream to write data to a destination; it may be a file, an array, peripheral device or socket.

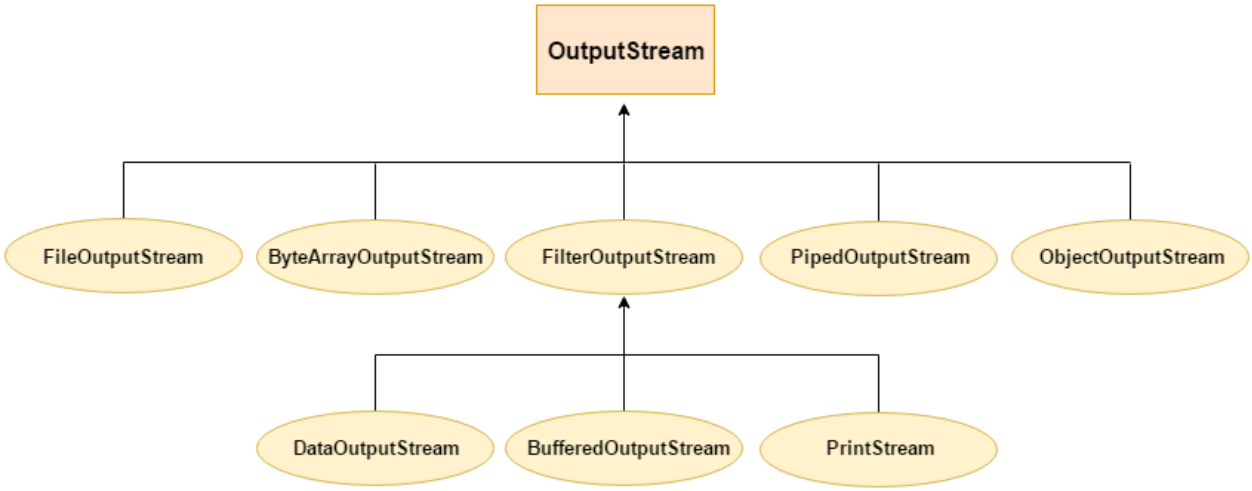
InputStream

Java application uses an input stream to read data from a source; it may be a file, an array, peripheral device or socket.

Let's understand the working of Java OutputStream and InputStream by the figure given below as-


1010101010
File
Console
Socket
1010101010
File
Console
Socket
InputStream
OutputStream
Source
1010101010
Read
Java Application
1010101010
Write
Destination


InputStream
FileInputStream
ByteArrayInputStream
FilterInputStream
PipedInputStream
ObjectInputStream
DataInputStream
BufferedInputStream
PushBackInputStream


OutputStream
FileOutputStream
ByteArrayOutputStream
FilterOutputStream
PipedOutputStream
ObjectOutputStream
DataOutputStream
BufferedOutputStream
PrintStream

Example-

1.Read a file line by line using Scanner class

```
import java.io.*;
import java.util.Scanner;

public class ReadLineByLineExample2 {
	public static void main(String args[]) {
		try {
//the file to be opened for reading
			FileInputStream fis = new FileInputStream("Demo.txt");
			Scanner sc = new Scanner(fis); // file to be scanned
//returns true if there is another line to read
			while (sc.hasNextLine()) {
				System.out.println(sc.nextLine()); // returns the line
that was skipped
			}
			sc.close(); // closes the scanner
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

Example- Write the file using File Writer class In this example, we are writing the data in the file testout.txt using Java FileWriter class.

```
package com.test
import java.io.FileWriter;
public class FileWriterExample {
   public static void main(String args[]){
       try{
         FileWriter fw=new FileWriter("D:\\testout.txt");
         fw.write("Velocity corporate training center pune.");
         fw.close();
        }catch(Exception e){System.out.println(e);}
        System.out.println("Success...");
    }
}
```