CORE JAVA

By : Kalpesh Chauhan

JAVA I/O

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. It's called a stream because it is like a stream of water that continues to flow.

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.

System.out.println("simple message");

System.err.println("error message");

Let's see the code to get **input** from console.

int i=System.in.read();//returns ASCII code of 1st character

System.out.println((**char**)i);//will print the character

OUTPUTSTREAM VS INPUTSTREAM

The explanation of OutputStream and InputStream classes are given below:

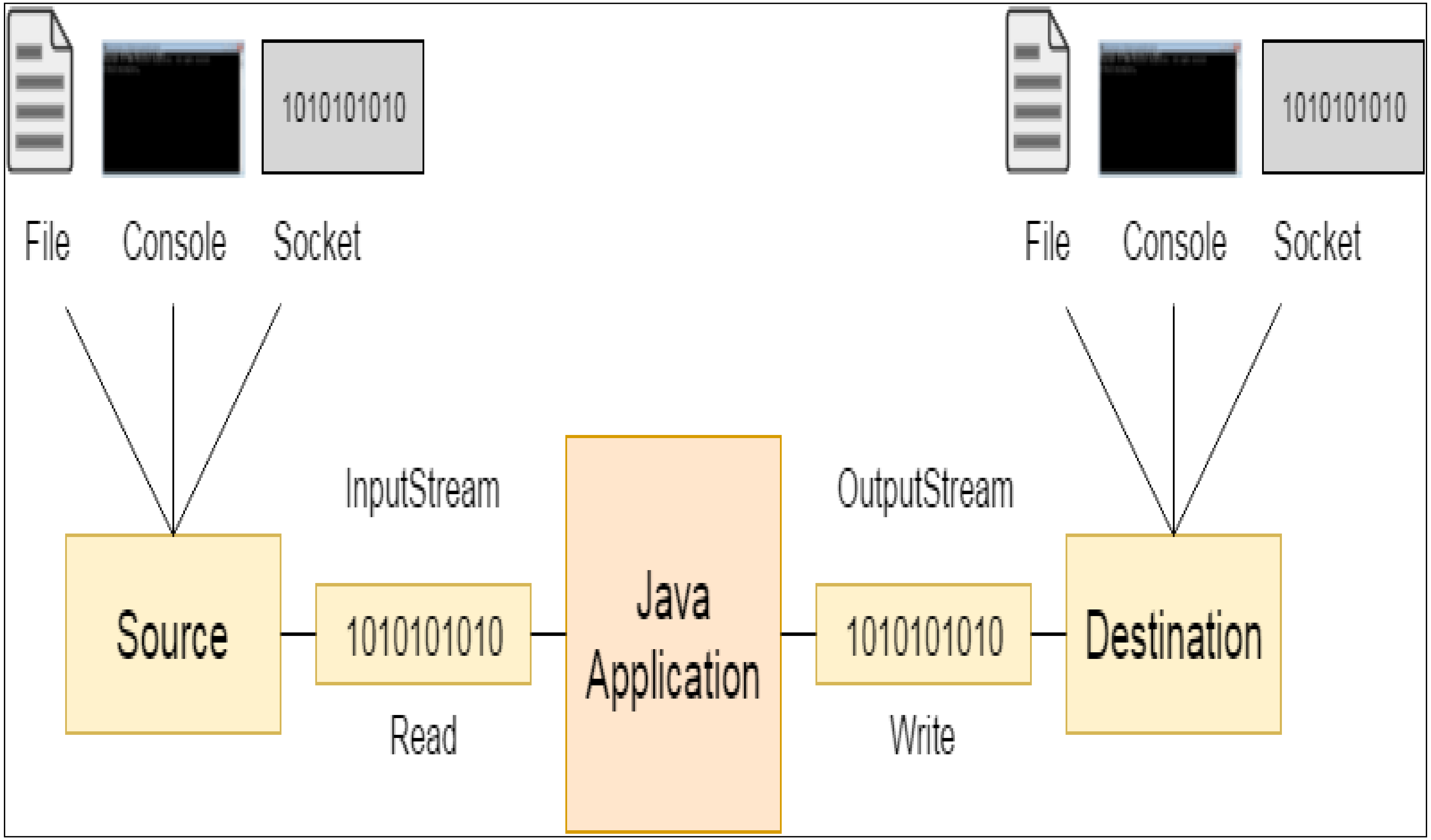
OutputStream

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.


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

OUTPUTSTREAM CLASS

OutputStream class is an abstract class. It is the superclass of all classes representing an output stream of bytes. An output stream accepts output bytes and sends them to some sink.

INPUTSTREAM CLASS

InputStream class is an abstract class. It is the superclass of all classes representing an input stream of bytes.

JAVA FILEOUTPUTSTREAM CLASS

Java FileOutputStream is an output stream used for writing data to a file.

If you have to write primitive values into a file, use FileOutputStream class. You can write byte-oriented as well as character-oriented data through FileOutputStream class. But, for character-oriented data, it is preferred to use FileWriter than FileOutputStream.

```
import java.io.FileOutputStream;
public class FileOutputStreamExample
{
        public static void main(String args[]){
                        try{

        FileOutputStream fout=new FileOutputStream("D:\\testout.txt");
                        fout.write(65);
                        fout.close();
                        System.out.println("success...");
                    }catch(Exception e){System.out.println(e);}
    }
}
```

```
import java.io.FileOutputStream;
public class FileOutputStreamExample {
    public static void main(String args[]){
            try{
                        FileOutputStream fout=new FileOutputStream("D:\\test.txt");
                        String s="Welcome to world of java.";
                        byte b[]=s.getBytes();//converting string into byte array
                        fout.write(b);
                        fout.close();
                        System.out.println("success...");
             }catch(Exception e){System.out.println(e);}
      }
}
```

JAVA FILEINPUTSTREAM CLASS

Java FileInputStream class obtains input bytes from a file. It is used for reading byte-oriented data (streams of raw bytes) such as image data, audio, video etc. You can also read character-stream data. But, for reading streams of characters, it is recommended to use FileReader class.

```
import java.io.FileInputStream;
public class DataStreamExample {
          public static void main(String args[]){
        try{
                  FileInputStream fin=new FileInputStream("D:\\test.txt");
                int i=0;
                  while((i=fin.read())!=-1)
                  {
                            System.out.print((char)i);
                  }
                  fin.close();
        }catch(Exception e){System.out.println(e);}
      }
  }
```

JAVA WRITER

It is an abstract class for writing to character streams. The methods that a subclass must implement are write(char[], int, int), flush(), and close(). Most subclasses will override some of the methods defined here to provide higher efficiency, functionality or both.

```
import java.io.*;
public class WriterExample {
            public static void main(String[] args) {
                        try {
                        Writer w = new FileWriter("output.txt");
                        String content = "I love my country";
                        w.write(content);
                        w.close();
                        System.out.println("Done");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

JAVA READER

Java Reader is an abstract class for reading character streams. The only methods that a subclass must implement are read(char[], int, int) and close(). Most subclasses, however, will override some of the methods to provide higher efficiency, additional functionality, or both.

```
import java.io.*;
public class ReaderExample {
        public static void main(String[] args) {
                try {
                        Reader reader = new FileReader("file.txt");
                        int data = reader.read();
                        while (data != -1) {
                                System.out.print((char) data);
                                data = reader.read();
                    }
                reader.close();
                } catch (Exception ex) {
                        System.out.println(ex.getMessage());
        }
  }
}
```

JAVA SCANNER CLASS

Java Scanner class comes under the java.util package. Java has various ways to read input from the keyboard, the java.util.Scanner class is one of them.

The Java Scanner class breaks the input into tokens using a delimiter that is whitespace by default. It provides many methods to read and parse various primitive values.

Java Scanner class is widely used to parse text for string and primitive types using a regular expression.

Java Scanner class extends Object class and implements Iterator and Closeable interfaces.

```
import java.util.*;
public class ScannerClassExample1 {
     public static void main(String args[]){
         System.out.println("--------Enter Your Details-------- ");
         Scanner in = new Scanner(System.in);
         System.out.print("Enter your name: ");
         String name = in.next();
         System.out.println("Name: " + name);
         System.out.print("Enter your age: ");
         int i = in.nextInt();
         System.out.println("Age: " + i);
         System.out.print("Enter your salary: ");
         double d = in.nextDouble();
         System.out.println("Salary: " + d);
         in.close();
         }
}
```

CONTINUE IN NEXT UNIT.....