Input & output in JAVA

How to Take Input From User in Java?

Java brings various Streams with its I/O package that helps the user perform all the Java input-output operations. These streams support all types of objects, data types, characters, files, etc. to fully execute the I/O operations. Input in Java can be with certain methods mentioned below in the article.

Methods to Take Input in Java

There are **two ways** by which we can take Java input from the user or from a file

- BufferedReader Class
- Scanner Class

1. Using BufferedReader Class for String Input In Java

It is a simple class that is used to read a sequence of characters. It has a simple function that reads a character another read which reads, an array of characters, and a readLine() function which reads a line.

InputStreamReader() is a function that converts the input stream of bytes into a stream of characters so that it can be read as BufferedReader expects a stream of characters. BufferedReader can throw checked Exceptions.

2. Using Scanner Class for Taking Input in Java

It is an advanced version of BufferedReader which was added in later versions of Java. The scanner can read formatted input. It has different functions for different types of data types.

- The scanner is much easier to read as we don't have to write throws as there is no exception thrown by it.
- It was added in later versions of Java
- It contains predefined functions to read an Integer, Character, and other data types as well.

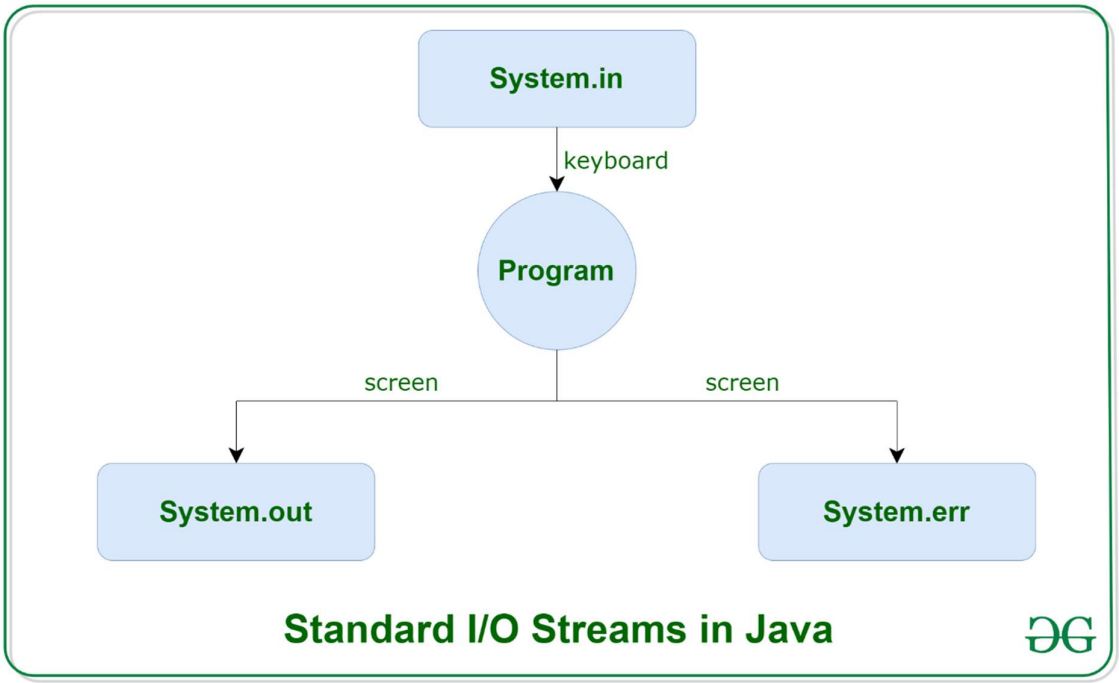
```
Scanner sc = new Scanner(System.in);
```

Now back to the deep knowledge about of I/O

This is GFG article.

Java brings various Streams with its I/O package that helps the user to perform all the input-output operations. These streams support all the types of objects, data-types, characters, files etc to fully execute the I/O operations.

Before exploring various input and output streams lets look at **3 standard or default streams** that Java has to provide which are also most common in use:



1. System.in: This is the **standard input stream** that is used to read characters from the keyboard or any other standard input device.

2. System.out: This is the **standard output stream** that is used to produce the result of a program on an output device like the computer screen.
   Here is a list of the various print functions that we use to output statements:

   - print(): This method in Java is used to display a text on the console. This text is passed as the parameter to this method in the form of String. This method prints the text on the console and the cursor remains at the end of the text at the console. The next printing takes place from just here.
     Syntax:
     ```
     System.out.print(parameter);
     ```
     Example:
     ```
     // Java code to illustrate print()
     import java.io.*;

     class Demo_print {
         public static void main(String[] args)
         {

             // using print()
             // all are printed in the
             // same line
             System.out.print("GfG! ");
             System.out.print("GfG! ");
             System.out.print("GfG! ");
         }
     }
     ```
     Output:
     ```
     GfG! GfG! GfG!
     ```
   - println(): This method in Java is also used to display a text on the console. It prints the text on the console and the cursor moves to the start of the next line at the console. The next printing takes place from the next line.
     Syntax:
     ```
     System.out.println(parameter);
     ```
     Example:

```java
// Java code to illustrate println()

import java.io.*;

class Demo_print {
    public static void main(String[] args)
    {

        // using println()
        // all are printed in the
        // different line
        System.out.println("GfG! ");
        System.out.println("GfG! ");
        System.out.println("GfG! ");
    }
}
```

Output:

```
GfG!
GfG!
GfG!
```

- printf(): This is the easiest of all methods as this is similar to printf in C. Note that System.out.print() and System.out.println() take a single argument, but printf() may take multiple arguments. This is used to format the output in Java.

3. **System.err:** This is the **standard error stream** that is used to output all the error data that a program might throw, on a computer screen or any standard output device.

   This stream also uses all the 3 above-mentioned functions to output the error data:

   - print()
   - println()
   - printf()