

I/O und GUI

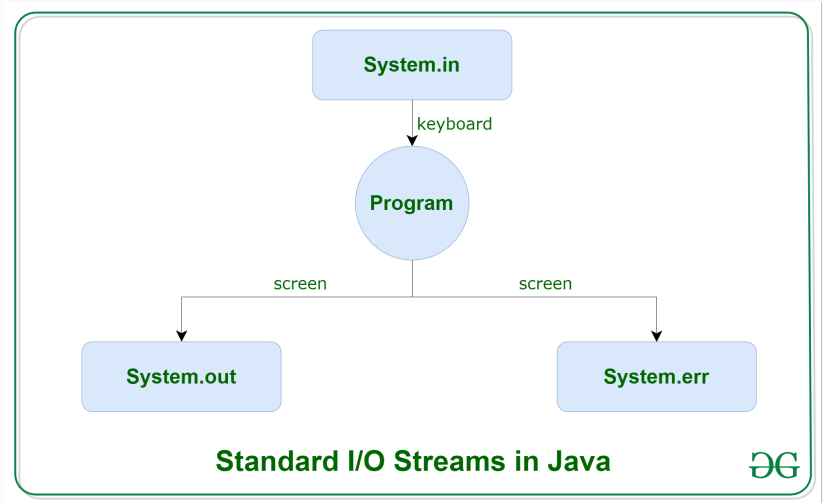
Konrad Raue, Oliver Scholz

14. Januar 2020

1. Input
2. Output
3. GUI
4. Nächste Woche

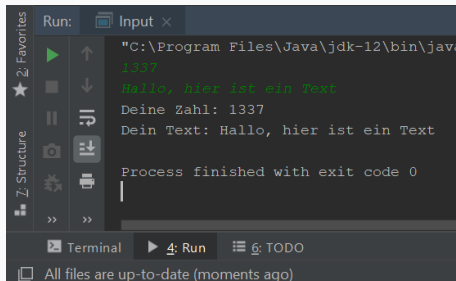
Input

Input und Output von Text



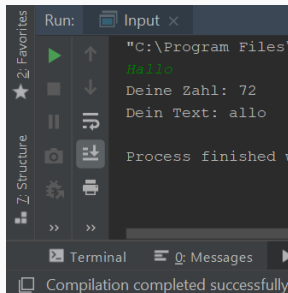
Input von Text

```
1 import java.io.BufferedReader;
2 import java.io.InputStreamReader;
3 public class Input{
4     public static void main(String[] args) throws Exception{
5         BufferedReader bufferedReader = new BufferedReader(
6             new InputStreamReader(System.in));
7         int number = Integer.parseInt(bufferedReader.readLine());
8         String text = bufferedReader.readLine();
9         System.out.println("Deine Zahl: "
10             + number + "\nDein Text: " + text);
11     }
12 }
```



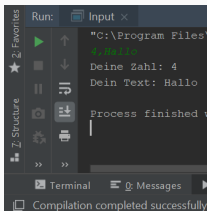
Input von Text

```
1 import java.io.BufferedReader;
2 import java.io.InputStreamReader;
3 public class Input{
4     public static void main(String[] args) throws Exception{
5         BufferedReader bufferedReader = new BufferedReader(
6             new InputStreamReader(System.in));
7         int number = bufferedReader.read();
8         String text = bufferedReader.readLine();
9         System.out.println("Deine Zahl: "
10             + number + "\nDein Text: " + text);
11     }
12 }
```



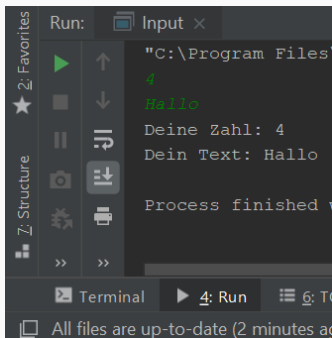
Input von Text

```
1 import java.io.BufferedReader;
2 import java.io.InputStreamReader;
3 import java.util.StringTokenizer;
4 public class Input{
5     public static void main(String[] args) throws Exception{
6         BufferedReader bufferedReader = new BufferedReader(
7             new InputStreamReader(System.in));
8         String input = bufferedReader.readLine();
9         StringTokenizer tokenizer = new StringTokenizer(input, ",");
10        int number = Integer.parseInt(tokenizer.nextToken());
11        String text = tokenizer.nextToken();
12        System.out.println("Deine Zahl: "
13            + number + "\nDein Text: " + text);
14    }
15 }
```



Input von Text

```
1 import java.util.Scanner;  
2 public class Input{  
3     public static void main(String[] args){  
4         Scanner scanner = new Scanner(System.in);  
5         int number = scanner.nextInt();  
6         String text = scanner.next();  
7         System.out.println("Deine Zahl: "  
8             + number + "\nDein Text: " + text);  
9     }  
10 }
```

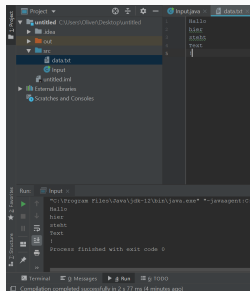


Einteilung in verschiedene Typen von Streams:

- <https://www.geeksforgeeks.org/java-io-input-output-in-java-with-examples/>

Input von Dateien

```
1 import java.io.FileInputStream;
2 public class Input{
3     public static void main(String[] args) throws Exception{
4         FileInputStream inputStream = new FileInputStream(
5             "src/data.txt");
6         int temp;
7         while((temp = inputStream.read()) != -1){
8             System.out.print((char) temp);
9         }
10        inputStream.close();
11    }
12 }
```



Output

Output von Text

```
1 public class Output{  
2     public static void main(String[] args){  
3         System.out.println("Hallo..");  
4         System.out.print("..noch mehr Text");  
5         int output = 48; //0  
6         System.out.print(output);  
7         System.out.write(output);  
8         System.out.flush();  
9     }  
10 }
```

Ausgabe:

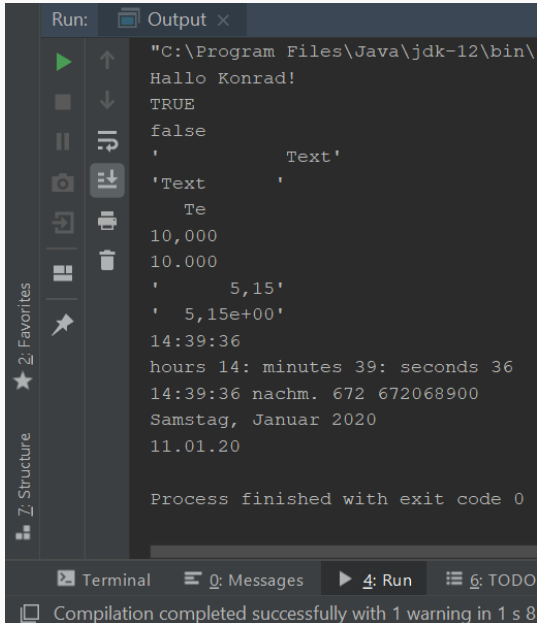
Hallo..

..noch mehr Text480

Output von Text

```
1 import java.time.LocalDateTime;
2 import java.util.Locale;
3 public class Output{
4     public static void main(String[] args){
5         System.out.printf("Hallo %s!\n", "Konrad");
6         System.out.printf("%B\n", 3);
7         System.out.printf("%b\n", null);
8         System.out.printf("'%15s'\n", "Text");
9         System.out.printf("'%-10s'\n", "Text");
10        System.out.printf("%5.2s", "Text");
11        System.out.printf(Locale.US, "%,d\n", 10000);
12        System.out.printf(Locale.GERMANY, "%,d\n", 10000);
13        System.out.printf("'%10.2f'\n", 5.1473);
14        System.out.printf("'%10.2e'\n", 5.1473);
15        LocalDateTime time = LocalDateTime.now();
16        System.out.printf("%tT\n", time);
17        System.out.printf("hours %tH: minutes %tM: seconds %tS\n",
18            time, time, time);
19        System.out.printf("%1$tH:%1$tM:%1$tS %1$tp %1$tL %1$tN %n",
20            time);
21        System.out.printf("%1$tA, %1$tB %1$tY %n", time);
22        System.out.printf("%1$td.%1$tm.%1$ty %n", time);
23    }
24 }
```

Output von Text



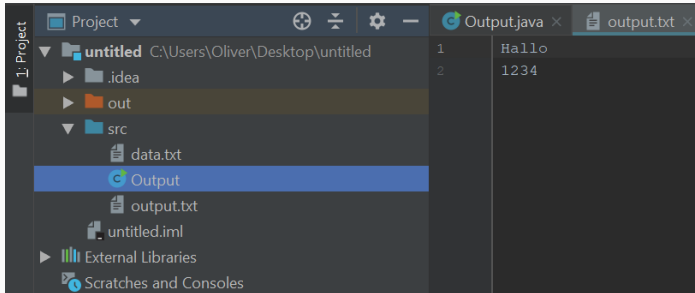
The screenshot shows an IDE's Run console. The title bar of the console window is "Run: Output x". The output text is as follows:

```
"C:\Program Files\Java\jdk-12\bin\  
Hallo Konrad!  
TRUE  
false  
'      Text'  
'Text  '  
    Te  
10,000  
10.000  
'      5,15'  
'  5,15e+00'  
14:39:36  
hours 14: minutes 39: seconds 36  
14:39:36 nachm. 672 672068900  
Samstag, Januar 2020  
11.01.20  
  
Process finished with exit code 0
```

On the left side of the console, there is a vertical toolbar with icons for Run (green play button), Stop (red square), Breakpoints (two horizontal lines), Run and Debug (bug icon), Run with Coverage (camera icon), Run with Profiler (magnifying glass icon), Run with Debugger (bug icon with magnifying glass), Run with Coverage and Profiler (camera icon with magnifying glass), and Run with Coverage and Profiler and Debugger (bug icon with magnifying glass and camera icon). Below the toolbar, there are tabs for "Z: Structure", "2: Favorites", and "★ 2: Favorites". At the bottom of the IDE, there is a status bar with tabs for "Terminal", "0: Messages", "4: Run", and "6: TODO". The status bar also displays the message "Compilation completed successfully with 1 warning in 1 s 8".

Output von Dateien

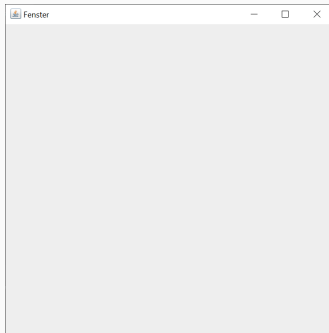
```
1 import java.io.FileOutputStream;
2 public class Output{
3     public static void main(String[] args) throws Exception{
4         FileOutputStream outputStream = new FileOutputStream("src/
5         output.txt");
6         String text = "Hallo\n1234";
7         for(int index = 0; index < text.length(); index++){
8             outputStream.write(text.charAt(index));
9         }
10        outputStream.close();
11    }
```



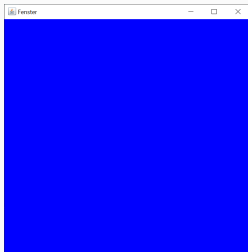
GUI

GUI

```
1 import javax.swing.*;  
2 public class GUI{  
3     public static void main(String[] args){  
4         JFrame window = new JFrame();  
5         window.setTitle("Fenster");  
6         window.setSize(500, 500);  
7         window.setVisible(true);  
8     }  
9 }
```

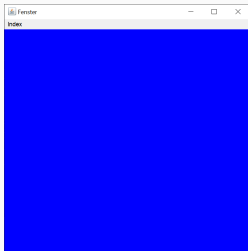


```
1 import javax.swing.*;
2 import java.awt.*;
3 public class GUI{
4     public static void main(String[] args){
5         ...
6         JPanel backgroundPanel = new JPanel();
7         backgroundPanel.setBackground(Color.BLUE);
8         window.add(backgroundPanel);
9     }
10 }
```



komplexere möglich (z.B. JSplitPane, JScrollPane, JTabbedPane)

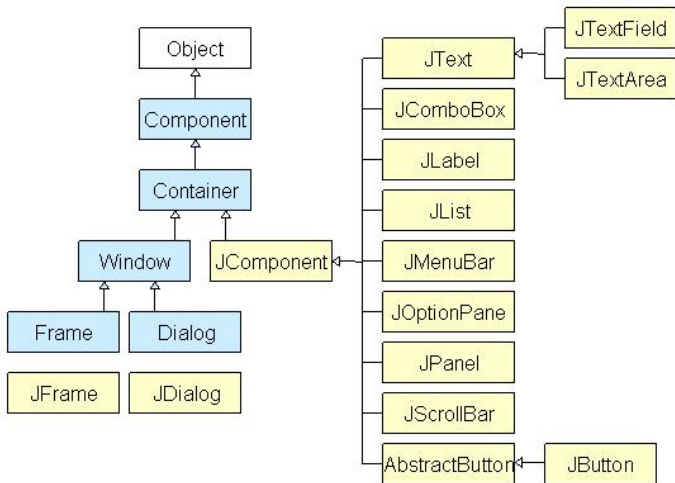
```
1 import javax.swing.*;
2 import java.awt.*;
3 public class GUI{
4     public static void main(String[] args){
5         ...
6         MenuBar bar = new MenuBar();
7         Menu menu = new Menu("Index");
8         MenuItem item = new MenuItem("Hallo");
9         menu.add(item);
10        bar.add(menu);
11        window.setMenuBar(bar);
12    }
13 }
```



```
1 import javax.swing.*;
2 import java.awt.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 public class GUI{
6     public static void main(String[] args){
7         ...
8         item.addActionListener(new ActionListener() {
9             @Override
10             public void actionPerformed(ActionEvent e) {
11                 System.out.println(e);
12             }
13         });
14         ...
15     }
16 }
```

```
1 import javax.swing.*;
2 import javax.swing.event.ListSelectionEvent;
3 import javax.swing.event.ListSelectionListener;
4 import java.awt.*;
5 import java.awt.event.ActionEvent;
6 import java.awt.event.ActionListener;
7 public class GUI{
8     public static void main(String[] args){
9         ...
10        JList myList = new JList();
11        Object[] values = new Object[10];
12        for(int i = 0; i < 10; i++){
13            values[i] = "Hallo " + i;
14        }
15        myList.setListData(values);
16        myList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
17        myList.addListSelectionListener(new ListSelectionListener() {
18            @Override
19            public void valueChanged(ListSelectionEvent e) {
20                System.out.println(myList.getSelectedIndex());
21            }
22        });
23        window.setContentPane(myList);
24        ...
25    }
26 }
```

Es lassen sich viele weitere Teile hinzufügen (z.B. Button, ButtonGroup).
Diese sind aus Swing (Java-GUI-Toolkit).
Hinzufügen: `window.getContentPane().add(myButton);`



weitere Beispiele: <https://www.guru99.com/java-swing-gui.html>

Nächste Woche

- Framework vs Library
- Spring