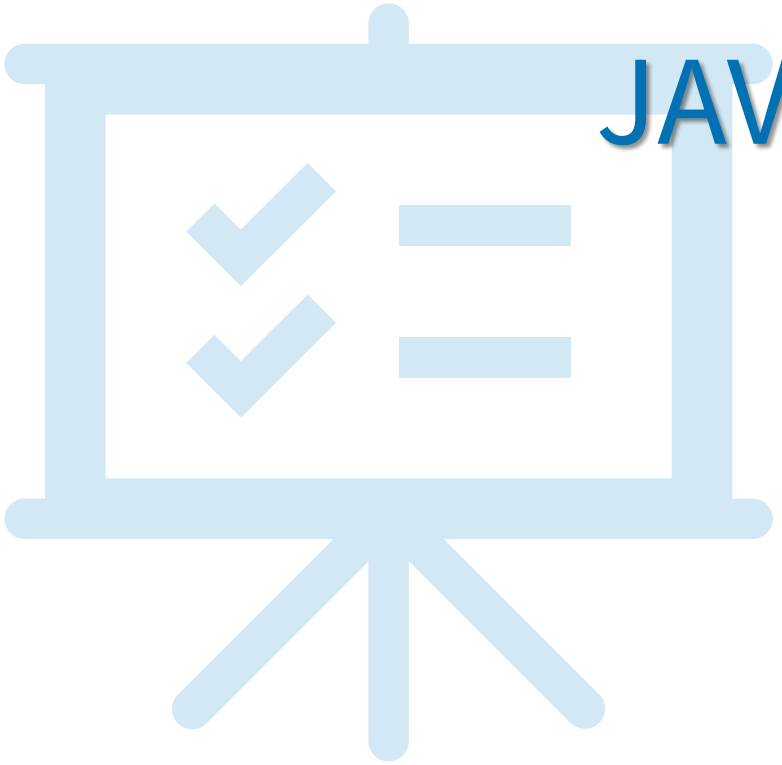


JAVA Programmierung

ECLIPSE & JAVA 8

Themenübersicht

01

Eingabe

1. Generelles
2. Beispiele

02

Aufgaben

1. Aufgabe
2. Lösung

03

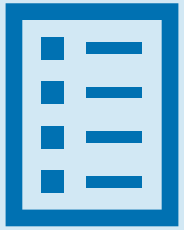
Konvertierung

1. Generelles
2. Beispiele

04

Aufgaben

1. Aufgabe
2. Lösung



01 Eingabe

1. *Generelles*
2. *Beispiele*

1.1 Eingabe

Generelles

1.1 Eingabe

- **Generelles**

Die Klasse „Scanner“ bietet die Möglichkeit mit dem Programm zu interagieren. Hierfür stellt die Klasse verschiedene Funktionalitäten bereit, die es dem Anwender ermöglichen, Texte, Zahlen, etc.. einzugeben. Am Ende muss der Scanner über die Funktion „close()“ wieder geschlossen werden.

=> **https://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html**

1.1 Eingabe

- **Generelles**

Der Scanner bietet unterschiedliche Methoden zum Auslesen der Konsole. Für fast jeden Datentyp gibt es eine Passende. In diesem Beispiel „nextInt()“.

1. Achtung ⚠
   Um mit einer weiteren Eingabe fortfahren zu können, muss der Stream auf das nächste Element ausgerichtet werden, sonst landen wir in einer Endlosschleife
2. Achtung ⚠
   Der Scanner muss am Ende immer geschlossen werden.

```java
// Scanner initialisierung => mit aufruf des Input Streams"System.in"
Scanner scanner = new Scanner(System.in);
int input;
boolean isKorrektForamt;
do {
    isKorrektForamt = false;
    try {
        System.out.println("bitte zahl eingeben");
        // Ließt die nächste Zeile der Konsole aus und speichert sie im input
        input = scanner.nextInt();
        // Sollte es kein int sein würde ein Fehler auftreten
    } catch (Exception ex) {
        // Dieser würde hier Gefangen
        // => in diesem Fall setzten wir den Boollean für die While Bedingung auf true
        // um das beenden der Schleife zu verhindern
        isKorrektForamt = true;
        // Wichtig ist hier den Stream für die nächste Eingabe vorzubereiten
        scanner.next();
    }
} while (isKorrektForamt);
scanner.close();
```

1.

2.

1.2 Eingabe

Beispiele

1.2 Eingabe

- **Beispiele**

```java
// Scanner initialisierung => mit aufruf des Input Streams"System.in"
Scanner scanner = new Scanner(System.in);
// das speichern der Eingabe in einem String
String inputString = scanner.next();
String inputString2 = scanner.nextLine();
// das speichern der Eingabe in einer numerischen Variable
int inputInt = scanner.nextInt();
// beim byte ist die Größe zu beachten
byte inputByte = scanner.nextByte();
long inputLong = scanner.nextLong();
// das speichern der Eingabe in einer decimal Variable
double inputDouble = scanner.nextDouble();
float inputFloat = scanner.nextFloat();
// boolean
boolean inputBool = scanner.nextBoolean();
// Zu jedem Datentyp gibt es immer noch die methode zum prüfen ob es
// weiter element gibt!
scanner.hasNext();
```

02 Aufgaben

1. *Aufgabe*
2. *Lösung*

2.1 Aufgaben

Aufgabe

2.1 Aufgabe

Erstellen Sie ein Programm welches den Scanner 2 mal verwendet und jeweils unterschiedliche Datentypen einliest. Nehmen Sie sich den Link zur Doku und recherchieren Sie geeignete Methoden.

2.2 Aufgabe

Lösung

2.2 Lösung

```
5    public static void main(String[] args) {
6        Scanner sc = new Scanner(System.in);
7
8        System.out.println("Gib einen Text ein: ");
9        String input1 = sc.nextLine();
10
11       System.out.println("Gib einen Integer-Wert ein: ");
12       int input2 = sc.nextInt();
13
14       System.out.println("Deine Eingaben: " + input1 + ",  " + input2);
15   }
16
```

03 Konvertierung

1. *Generelles*
2. *Beispiele*

3.1 Konvertierung

Generelles

3.1 Konvertierung

- **Generelles**

Java stellt über die Wrapper-Klassen der jeweiligen Datentypen für fast alle eine geeignete Parser Methode bereit.
Diese sollte immer innerhalb eines Try-Catch-Blockes stehen, um etwaige Fehler abzufangen.

Generell unterscheidet man zwischen zwei Arten des Konvertierung

- **Implizite Konvertierung von primitiven Datentypen**
- **Explizite Konvertierung (cast) von primitiven Datentypen**

3.1 Konvertierung

- **Implizite Konvertierung von primitiven Datentypen**

Je nach Wertigkeit der Variable kann eine Typumwandlung relative einfach aussehen. Zu beachten ist allerdings, daß es zu Wertverlust kommen kann. Zum Beispiel von Float nach int. Generell lässt sich sagen das der kleiner Typ immer in den nächst größeren Typ konvertiert werden kann. Also implizit (d. h. ohne "casting")

byte	=>	short	=>	int	=>	long	=>	float	=>	double	
int	=>	char									

Hinweis:
Ein numerischer Wert, sofern er in der Ascii Tabelle aufgeführt wird, kann in einen Char (und umgekehrt) konvertiert werden kann.

3.1 Konvertierung

- **Explizite Konvertierung (cast) von primitiven Datentypen**

Eine explizite Konvertierung (mal von boolean abgesehen) funktioniert immer. Dabei wird der zu Konvertierenden Variable der Typ der neunen Variable innerhalb der runden Klammern voran gestellt.

```
int konvertInt = 10;
byte konvertByte = (byte)konvertInt;
```

Hierbei ist zu beachten, das es zu unerwarteten Abweichungen führen kann, sollte die Wertigkeit die maximale Größe der neuen Variable überschreiten.

```
int konvertInt = 1234567891;
byte konvertByte = (byte)konvertInt;
System.out.println(konvertByte);
```

```
<terminated> FirstClass [Java A
-45
```

3.2 Konvertierung

Beispiele

3.2 Konvertierung

- **Beispiel**

```
5   public static void main(String[] args) {
6       try {
7           int x = Integer.parseInt("9");
8           int b = Integer.parseInt("444", 16);
9           double c = Double.parseDouble("5");
10          boolean result = Boolean.parseBoolean("TrUe");
11      } catch (Exception e) {
12          System.out.println(e.getMessage());
13      }
14
```

Radix Parameter

- 2 - Basis-2-Zahlen (Binär). 2 bedeutet, dass der Eingabewert nur 0-1 enthält. Andernfalls => NumberFormatException
- 8 - Basis-8-Zahlen (Oktal). 8 bedeutet, dass die Eingabezeichenfolge nur 0-7 enthält. Andernfalls => NumberFormatException
  10 - Zahlen zur Basis 10 (Dezimal). Es kann 0-9 Zeichen enthalten.
- 16 - Zahlen zur Basis 16 (Hexa-Dezimalzahl). Es kann 0-9 und die Zeichen A, B, C, D, E, F enthalten.
- 36 - Zahlen zur Basis 36 (Zahlen und Buchstaben). Es kann 0-9 (10) Zahlen und A bis Z (26) Buchstaben enthalten.

04 Aufgaben

1. *Aufgabe*
2. *Lösung*

4.1 Aufgaben

Aufgabe

4.1 Aufgabe

Erweitern Sie das erste Programm. Diesmal soll die Eingabe einen Parser benutzen…
Seien Sie kreativ.

Beschäftigen Sie sich mit dem Thema „Explizites und implizites Konvertieren“

4.1 Aufgaben

Lösung

4.2 Lösung

```
5      public static void main(String[] args) {
6          Scanner sc = new Scanner(System.in);
7          int input1 = 0;
8          System.out.println("Gib einen Integer-Wert ein: ");
9          try {
10             input1 = Integer.parseInt(sc.nextLine());
11         } catch (Exception e) {
12             System.out.println(e.getMessage());
13         }
14
15         System.out.println("Deine Eingabe: " + input1);
16     }
17
18 }
```

VIELEN DANK!