



Variablen (1/2)

openHPI-Java-Team
Hasso-Plattner-Institut



- Container für Daten
- Benötigen einen eindeutigen Variablennamen
- Format: <Datentyp> <Bezeichner>;

String output;

↑ ↑

Datentyp Bezeichner



Wiederverwendung von Variablen

```
1  class HelloPaco{
2      public static void main(String[] args){
3          String output;
4          output = "Hallo Paco";
5          System.out.println(output);
6          System.out.println(output);
7      }
8  }
```

Ausgabe:

Hallo Paco

Hallo Paco



Variablen Kurzschreibweise

- Container für Daten
- Benötigen einen eindeutigen Variablennamen
- Format: <Datentyp> <Bezeichner> = <Wert>;

```
String output = "Hallo Paco";
```

↑ ↑ ↑

Datentyp Bezeichner Wert



Wiederverwendung von Variablen

```
1  class HelloPaco{
2      public static void main(String[] args){
3          String output = "Hallo Paco";
4          System.out.println(output);
5          System.out.println(output);
6      }
7  }
```

Ausgabe:

Hallo Paco

Hallo Paco



Neuzuweisung von Variablen

```
1  class HelloPaco{
2      public static void main(String[] args){
3          → String output = "Hallo Paco";
4              System.out.println(output);
5          → output = "Hallo Duke";
6              System.out.println(output);
7      }
8  }
```

Datentyp

- wird nur initial angegeben
- Wird nach der Initialisierung nicht mehr hingeschrieben
 - Initialisieren ist das Anlegen einer Variable



Neuzuweisung von Variablen

```
1  class HelloPaco{
2      public static void main(String[] args){
3          String output = "Hallo Paco";
4          System.out.println(output);
5          output = "Hallo Duke";
6          System.out.println(output);
7      }
8  }
```

Ausgabe:

Hallo Paco

Hallo Duke



Name	Art	Beispiele
String	Text, Zeichenkette	String name = "Robin";
char	Buchstabe, Zeichen	char country = 'd'; char cedille = 'ç';
int	Ganzzahl	int age = 2; int truth = -42;
double	Kommazahl	double speed = 98.7;



Verkettung von Datentypen (1/4)

```
1 void print(){  
2     String output = "Hallo Paco";  
3     System.out.println(output + "!");  
4 }
```

Strings

- Werden mit + zusammengefügt



Verkettung von Datentypen (2/4)

```
1 void print(){  
2     String output = "Hallo Paco";  
3     System.out.println(output + "!");  
4 }
```

Ausgabe:

Hallo Paco!



Verkettung von Datentypen (3/4)

```
1 void print(){  
2     String welcome = "Hallo";  
3     String name = "Paco";  
4     System.out.println(welcome + " " + name);  
5 }
```

Formatierung

- Auf Leerzeichen zwischen Wörtern achten



Verkettung von Datentypen (4/4)

```
1 void print(){
2     String welcome = "Hallo";
3     String name = "Paco";
4     System.out.println(welcome + " " + name);
5 }
```

Ausgabe:

Hallo Paco



Variablen (1/2)

openHPI-Java-Team
Hasso-Plattner-Institut



Variablen (2/2)

openHPI-Java-Team
Hasso-Plattner-Institut



Verkettung von Datentypen (1/2)

```
1 void print(){  
2     String first = "5";  
3     String second = "5";  
4     System.out.println(first + second);  
5 }
```

Ausgabe:

55



Verkettung von Datentypen (2/2)

```
1 void print(){
2     int number1 = 5;
3     int number2 = 5;
4     System.out.println(number1 + number2);
5 }
```

Ausgabe:

10

Vorsicht!

- Addition bei Integers
- Konkatination bei Strings

Eine Auswahl an Datentypen



Name	Art	Beispiele
String	Text, Zeichenkette	String name = "Robin";
char	Buchstabe, Zeichen	char country = 'd'; char cedille = 'ç';
int	Ganzzahl	int age = 2; int truth = -42;
double	Kommazahl	double speed = 98.7;



```
1 void printCases(){
2     int solvedCases = 0;
3     int unsolvedCases = 1;
4     int numberOfCases = solvedCases + unsolvedCases;
5     System.out.println(numberOfCases);
6 }
```

Ausgabe:

1



```
1 void printCases(){  
2     int solvedCases = 0;  
3     int unsolvedCases = 1;  
4     System.out.println(solvedCases + unsolvedCases);  
5 }
```

Formatierung

- Kein Semikolon innerhalb der Klammern

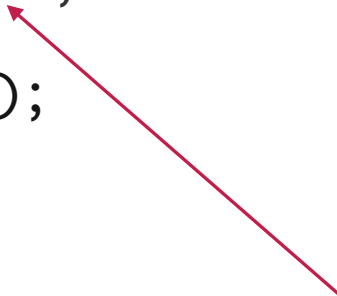
Ausgabe:

1



Verschiedene Datentypen

```
1 void print(){  
2     String text = "Ich mag die Zahl ";  
3     int number = 7;  
4     System.out.println(text + number);  
5 }
```



Java Ausgabe

- Erlaubt auch verschiedene Datentypen
- Tipp: Leerzeichen mit in den String (`text`) schreiben

Ausgabe:

Ich mag die Zahl 7



```
1 void calculate(){
2     int number1 = 42;
3     int number2 = 21;
4     System.out.println(number1 - number2);
5     System.out.println(number1 / number2);
6     System.out.println(number2 * 2);
7 }
```

Ausgabe:

21

2

42

Operatoren der Grundrechenarten bei `int`



Operator	Operation
+	Addition
-	Subtraktion
*	Multiplikation
/	Division (ganzzahliger Anteil bei <code>int</code>)
%	Modulo (Rest der Division bei <code>int</code>)



```
1 void calculate(){
2     int number1 = 0;
3     number1 = number1 + 2;
4
5 }
```

A red bracket is drawn under the expression 'number1 + 2' on line 3, indicating it is evaluated first. A red arrow then points from the right side of the bracket down to the left side of the assignment operator '=', showing the flow of the evaluation from right to left.

Auswertungsreihenfolge

- Es wird immer erst die rechte Seite ausgewertet
- Und dann der linken Seite zugewiesen



```
1      calculate(){
2      int number1 = 0;
3      number1 = number1 + 2;
4      System.out.println(number1);
5  }
```

Ausgabe:

2



```
1 void calculate(){
2     int number1 = 22;
3     number1 = number1 + 2;
4     System.out.println(number1);
5 }
```

Ausgabe:

24



```
1 void calculate(){
2     int number = 0;
3     number = number + 2;
4     number = number + 1;
5 }
```

```
1 void calculate(){
2     int number = 0;
3     number += 2;
4     number++;
5 }
```

Operation	Kurzschreibweise
Addition	+=
Subtraktion	-=
Multiplikation	*=
Division	/=
Inkrementieren um 1	++
Dekrementieren um 1	--

Division bei `int`



```
1 void calculate(){
2     int number1 = 5;
3     number1 /= 2;
4     System.out.println(number1);
5 }
```

Ausgabe:

2

Achtung! Die Division bei `int` errechnet nur den ganzzahligen Anteil



```
1 void calculate(){
2     int number1 = 5;
3     number1 = number1 % 2;
4     System.out.println(number1);
5 }
```

Ausgabe:

1

Achtung! Der Modulo Operator liefert den Rest der ganzzahligen Division

$$5 = 2 * 2 + 1$$



Name	Art	Beispiele
String	Text, Zeichenkette	String name = "Robin";
char	Buchstabe, Zeichen	char country = 'd'; char cedille = 'ç';
int	Ganzzahl	int age = 2; int truth = -42;
double	Kommazahl	double speed = 98.7;

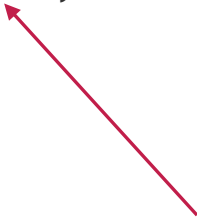


Operator	Operation
+	Addition
-	Subtraktion
*	Multiplikation
/	Division



Rechnen mit Fließkommazahlen

```
1 void calculate(){  
2     double weight = 42.2;  
3     weight--;  
4     weight *= 2;  
5 }
```



Fließkommazahlen

- Werden in Java zum Beispiel durch `double` dargestellt
- **Achtung!** Java nutzt einen Punkt statt einem Kommazeichen um Vor- und Nachkommastellen zu trennen
 - Also 3.1415 statt 3,1415



Division bei `double`

```
1 void calculate(){
2     double number1 = 5;
3     number1 /= 2;
4     System.out.println(number1);
5 }
```

Ausgabe:

2.5



Rechnen mit Fließkommazahlen

```
1 void calculate(){  
2     double number1 = 0.3;  
3     double number2 = 0.1;  
4     System.out.println(number1 - number2);  
5 }
```

Ausgabe:

0.19999999999999998

Vorsicht! bei Verwendung von `double` und anderen Fließkommazahlen



Variablen (2/2)

openHPI-Java-Team
Hasso-Plattner-Institut