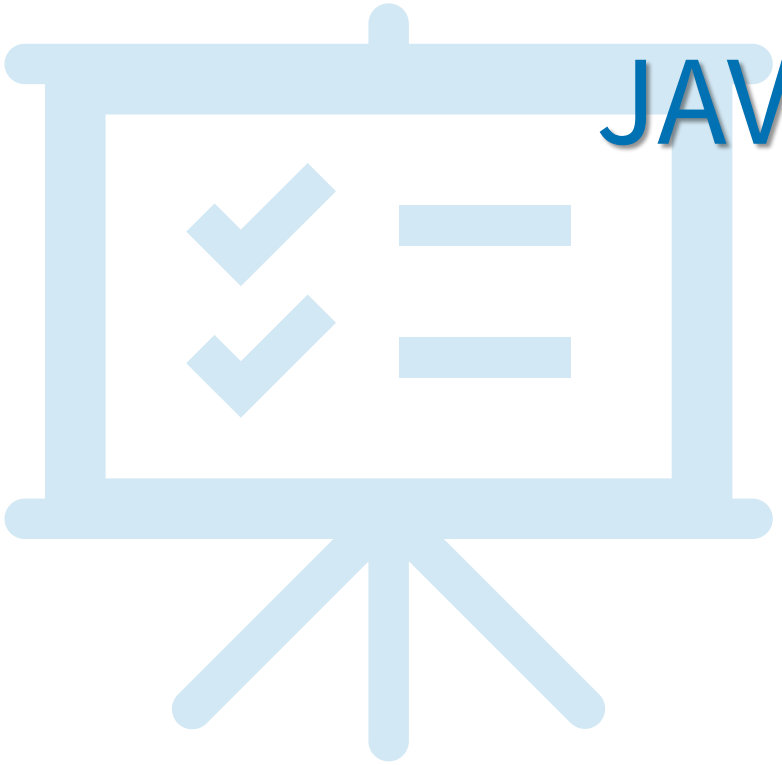
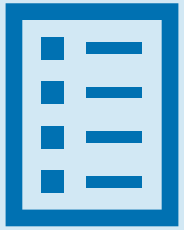


JAVA Programmierung

ECLIPSE & JAVA 8

Themenübersicht



01

String

1. Generelles
2. Beispiele

02

Aufgabe

1. Aufgabe
2. Lösung

03

Wrapperklassen

1. Generelles
2. Beispiele

04

Aufgabe

1. Aufgabe
2. Lösung

01 String

1. *Generelles*
2. *Beispiele*

1.1 String

Generelles

1.1 String

- **Generelles**

Technisch gesehen ist ein String in Java eine Aneinanderreihung von Einzelbuchstaben (Datentyp char – ein einzelner Buchstabe). String ist eine Klasse. Deshalb wird String auch im Gegensatz zu int oder double groß geschrieben. Der Speicherplatz eines Strings kann unterschiedlich groß sein. Eine String-Variable enthält nicht direkt die Zeichenkette, sondern ist eine Referenz, also ein Verweis auf eine Adresse im Speicher, bei der die Zeichenkette beginnt.
Eine String-Variable kann auch mit null besetzt sein. Das bedeutet, dass die Referenz auf keinen String zeigt.

Eine Zuweisung funktioniert auf die gleiche Weise wie bei einer primitiven Variablen.

⇒ https://docs.oracle.com/javase/7/docs/api/java/lang/String.html

1.2 String

Beispiele

1.2 String

- **Beispiel**

```
5      public static void main(String[] args) {
6          // Zeichen extrahieren per charAt() und toCharArray()
7          String test = "Dies ist ein Test";
8          System.out.println(test.charAt(7)); // Ausgabe: t
9          char[] c = test.toCharArray();
10         System.out.print(c[0]); // Ausgabe: D
11         System.out.print(c[1]); // Ausgabe: i
12         System.out.print(c[2]); // Ausgabe: e
13         char letztesZeichen = test.charAt(test.length() - 1);
14         System.out.println(letztesZeichen);
15
```

```
String s1 = " 123 ";
System.out.println(s1.replace('2', '7')); // Ausgabe: " 127 "
System.out.println(s1.replace("23", "77")); // Ausgabe: " 177 "
System.out.println(s1.trim()); // Ausgabe: "123"
```

1.2 String

- **Beispiel**

```
// Werte/Integer in Strings wandeln per Integer.toString()
int i = 10;
String test3 = "";
test3 = "" + i; // simple/harte Methode
System.out.println(test3);
test3 = Integer.toString(i); // besser
System.out.println(test3);
```

```
// Werte in Strings wandeln mit valueOf()
String test2 = "";
test2 = String.valueOf('x');
System.out.println(test2); // Ausgabe: x
test2 = String.valueOf(42);
System.out.println(test2); // Ausgabe: 42
```

```
// Teilstrings extrahieren per substring()
String test4 = "Dies ist ein Test";
String test5 = test.substring(9); // Teilstring von Zeichen 9 bis Rest
String test6 = test.substring(0, 4); // Teilstring von Anfang bis Zeichen 4
System.out.println(test5); // Ausgabe: ein Test
System.out.println(test6); // Ausgabe: Dies
```

1.2 String

- **Beispiel**

```
// Strings vergleichen per equals() und equalsIgnoreCase()
String s = "test";
System.out.println(s.equals("test")); // Ausgabe: true
System.out.println(s.equals("TEST")); // Ausgabe: false
System.out.println(s.equalsIgnoreCase("TEST")); // Ausgabe: true

// Zeichen ersetzen/entfernen per replace() oder trim()
String s2 = " 121 ";
System.out.println(s2.replace('2', '3')); // Ausgabe: 131
// Das Ersetzen von Teilstrings durch Teilstrings ist nicht möglich
System.out.println(s2.trim()); // Ausgabe: 121
```

02 Aufgaben

1. *Aufgabe*
2. *Lösung*

2.1 Aufgaben

Aufgabe

2.1 Aufgabe

Zerlegen Sie den String „Hallo Welt“ in seine Einzelteile.

- **Geben Sie diese Einzelteile aus**
- **Zählen Sie die Einzelteile und geben Sie das Ergebnis aus**
- **Fügen Sie die Einzelteile wieder zusammen**

Suchen Sie hierfür die geeigneten Methoden in der JavaDoc raus.

2.1 Aufgaben

Lösung

2.2 Lösung

Zerlegen Sie den String „Hallo Welt“ in seine Einzelteile.

- **Geben Sie diese Einzelteile aus**
- **Zähle Sie die Einzelteile und geben Sie das Ergebnis aus**
- **Fügen Sie die Einzelteile wieder zusammen**

```
String test = "Hallo Welt";
System.out.println(test.length()); // Ausgabe: 10
String[] testArray = test.split("");
String newString = "";
for(String c : testArray) {
    System.out.println(c);
    newString += c;
}
System.out.println(newString);
```

03 Wrapperklassen

1. *Generelles*
2. *Beispiele*

3.1 Wrapperklassen

Generelles

3.1 Wrapperklassen

• Generelles

Zu jedem Datentypen gibt es eine dazugehörige Wrapperklasse. Diese stellen im übertragenden Sinne ein Hülle für die primitiven Typen dar. Daher können wir auch für primitive Datentypen alle Vorteile der Objekt-Orientierten Programmierung nutzbar machen.

Wrapper Objekte werden über den „new“ Operator erstellt. Momentan ist es wichtig zu verstehen, das wir nur so Objekte der Wrapper Klassen erstellen können.

3.1 Wrapperklassen

- **Generelles**

Primitiver Datentyp	Wrapper-Klasse	Konstruktor-Argumente
boolean	Boolean	boolean oder String
byte	Byte	byte oder String
char	Character	char
double	Double	double oder String
float	Float	float, double oder String
int	Integer	int oder String
long	Long	long oder String
short	Short	short oder String

3.1 Wrapperklassen

Beispiele

3.2 Wrapperklassen

- **Beispiel**

```
int zahl = 500;
Integer zahlInteger = new Integer(zahl);
System.out.println("\n" + zahlInteger);  // Ausgabe 500
System.out.println("\n" + zahlInteger.getClass()); // Ausgabe class java.lang.Integer
System.out.println("\n" + zahlInteger.getClass().getSuperclass());  // Ausgabe class java.lang.Number

int zahld = 500;
Double zahlDouble = new Double(zahld);
System.out.println("\n" + zahlDouble);  // Ausgabe 500.0
System.out.println("\n" + zahlDouble.getClass());  // Ausgabe class java.lang.Double
System.out.println("\n" + zahlDouble.getClass().getSuperclass());  // Ausgabe class java.lang.Number
```

3.2 Wrapperklassen

- **Beispiel**

```
int zahlf = 500;
Float zahlFloat = new Float(zahlf);
System.out.println("\n" + zahlFloat);  // Ausgabe 500.0
System.out.println("\n" + zahlFloat.getClass());  // Ausgabe java.lang.Float
System.out.println("\n" + zahlFloat.getClass().getSuperclass()); // Ausgabe class java.lang.Number

String zahl1 = "501";
String zahlString = new String(zahl1);
System.out.println("\n" + zahlString);  // Ausgabe 501
System.out.println("\n" + zahlString.getClass()); // Ausgabe class java.lang.String
System.out.println("\n" + zahlString.getClass().getSuperclass());  // Ausgabe class java.lang.Object

Boolean bool1 = new Boolean("trUe");
System.out.println("\n" + bool1);  // Ausgabe true
System.out.println("\n" + bool1.getClass()); // Ausgabe class java.lang.Boolean
System.out.println("\n" + bool1.getClass().getSuperclass());  // Ausgabe class java.lang.Object
```

04 Aufgaben

1. *Aufgabe*
2. *Lösung*

4.1 Aufgaben

Aufgabe

4.1 Aufgabe

Schauen Sie sich die Methoden und Konstruktoren der jeweiligen Wrapperklassen an und initialisieren Sie Objekte damit.

4.2 Aufgaben

Lösung

4.2 Lösung

```
20
21    public static void main(String[] args) {
22        int zahl = 500;
23        String zahlString = "501";
24        Integer zahlInteger = new Integer(zahl);
25        System.out.println("\n" + zahlInteger);
26        System.out.println(zahlInteger.intValue());
27        Integer zahlInteger2 = new Integer(zahlString);
28        System.out.println("\n" + zahlInteger2.intValue());
29
30        String intString = "501";
31        String byteString = "127";
32        int zahlInt = Integer.parseInt(intString);
33        System.out.println("\n" + zahlInt);
34
35        byte zahlByte = Byte.parseByte(byteString);
36        System.out.println(zahlByte);
37
38        Boolean bool = new Boolean("True");
39        System.out.println(bool);
40        Short sh = new Short("1");
41        System.out.println(sh);
42    }
43 }
```

```
<terminated> test [Java A

500
500

501

501
127
true
1
```

VIELEN DANK!