UNIVERSITÄT
LEIPZIG

Einführung in die Objekt-Orientierte Modellierung und Programmierung

Wintersemester 2025/2026

Dirk Zeckzer

Institut für Informatik

Teil VII

Ausdrücke

Java: Arithmetische Operatoren

`+, -, *, /, %`

`a = b + c;`	int	a, b, c;	double	a, b, c;	
`a = b - c;`	int	a, b, c;	double	a, b, c;	
`f = g * h;`	int	f, g, h;	double	f, g, h;	
`a = b / c;`	int	a, b, c;			div
`a = b % c;`	int	a, b, c;			mod
`f = g / h;`			double	f, g, h;	

```java
int a = 32;
int b = 10;

int c = a / b;
// Ergebnis: 3
int d = a % b;
// Ergebnis: 2
```

```java
double g = 32;
double h = 10;

double f = g / h;
// Ergebnis: 3.2
```

Java: Arithmetische Operatoren

Kurzschreibweisen

`+=, -=, ...`

```
a = a + b;   ⟹   a += b;
```

`++, --`

```
a = a + 1;  ⇔  a += 1;  ⇔  ++a;  ⇔  a++;
a = a - 1;  ⇔  a -= 1;  ⇔  --a;  ⇔  a--;
```

Java: Arithmetische Operatoren und Type Casts

- Arithmetische Operatoren für denselben primitiven Datentyp für
  - `long`: analog `int`
  - `float`: analog `double`

- Arithmetische Operatoren für denselben primitiven Datentyp für
  - `byte`
  - `short`

  werden vor Berechnungen in `int` umgewandelt
  (impliziter **type cast**)
- Müssen anschließend in den Ziel-Datentyp umgewandelt werden
- Bei +=, ... wird automatisch gecastet

Java: Arithmetische Operatoren Beipiele

```
1  int i = 7 + 5 % 4;
```

$\rightarrow 7 + 5 \,\%\, 4 = 7 + 1 = 8$
als 32 bit ganze Zahl

```
1  long l = 5L * 7L + 4L;
```

$\rightarrow 5 * 7 + 4 = 35 + 4 = 39$
als 64 bit ganze Zahl

Java: Arithmetische Operatoren und Type Casts

```java
byte myByte1 = 1;
byte myByte2 = 1;
byte myByte;
myByte = (byte) (myByte1 + myByte2);
```

```java
short myShort1 = 1;
short myShort2 = 1;
short myShort;
myShort = (short) (myShort1 + myShort2);
```

```java
short myShort2 = 1;
short myShort;
myShort -= myShort2; // Kein cast notwendig
```

Java: Arithmetische Operatoren und Type Casts

```java
long l = 5;  // int -> long
double d1 = 7.0f;  // float -> double
double d2 = 1;  // int -> double
float f = -24L;  // long -> float
```

```java
int i = 7;
long l = 5;
double d1 = 2.0;
int i = (int) ((i + l) / d1);
```

$\rightarrow$

```java
int i = (int) (((double) (((long) i) + l)) / d1);
```

(mit impliziten casts)

Java: Arithmetische Operatoren und Type Casts

```java
double d1 = 1 / 2;
// -> 0.0
double d2 = (double) 1 / 2;
// Definiert, will ich mir nicht merken
double d3 = 1.0 / 2.0;
// -> 0.5, das will ich berechnen
```

```java
int sieben = 7;
int zehn = 10;
double d1 = sieben / zehn;
// -> 0.0
double d2 = (double) sieben / zehn;
// Definiert, will ich mir nicht merken
double d3 = ((double) sieben) / ((double) zehn);
// 0.7, das will ich berechnen
```

Java: Vergleiche

Mathematische Vergleiche

```
==, !=, <, >, <=, >=
```

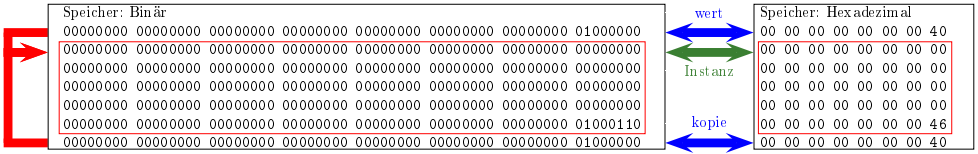
```
boolean b; int a,c;
b = (a == c);
```

Vergleiche von Instanzen

- `==` identische Instanzen
- `!=` unterschiedliche Instanzen (können gleiche Attribut-Belegung haben!)

Java: Speicherverwaltung

- Vergleich von Instanzen
  - `Integer wert = new Integer(70);`
  - `Integer kopie = wert;`
  - `kopie == wert` → `true`



Java: Speicherverwaltung

- Vergleich von Instanzen
  - `Integer wert1 = new Integer(70);`
  - `Integer wert2 = new Integer(70);`
  - `wert1 == wert2` → `false`

Java: Logische Operatoren

`&&` : logisches und

`||` : logisches oder

`!` : logisches nicht

```
(!( a == b )) && (d > e)
```