JAVA SPRACHKONSTRUKTE

Operatoren und Ausdrücke

Operatoren und Ausdrücke

- Operatoren ermöglichen das Rechnen und Verändern von in Variablen und Attributen gespeicherten Werten.
- Java unterstützt
  - *arithmetische Operatoren*
  - *logische Operatoren*

ARITHMETISCHE OPERATOREN

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel	Art

Inkrement;
Erhöht eine Variable um den Wert 1
	++	Ganze Zahlen,
		Fließkommazahlen
			int a = 3;
			a++;
			float b = 3f;
			b++;
				Unär

Arithmetische Addition;
Das Ergebnis der Berechnung entsprich dem des Operanden mit dem größten Wertebereich
z.B. int+int = int
int + long = long
	+	Ganze Zahlen,
		Fließkommazahlen
			int c,d,e;
			c = 3;
			d = 5;
			e = c + d;
			
			int x;
			long y,z;
			x = 3;
			y = 4;
			z = x + y;
				Binär

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel	Art

Arithmetische Subtraktion	-	Ganze Zahlen, Fließkommazahlen	int e = 3;
			float f = 4;
			float g;
			g = e – f;
				Binär

Arithmetische Multiplikation	*	Ganze Zahlen, Fließkommazahlen	int c,d;
			c = 3;
			d = c * 4;
				Binär

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel	Art

Arithmetische Division; berechnet Quotienten aus Dividend und Divisor	/	Ganze Zahlen: Sind beide Operanden ganze Zahlen, wird das Ergebnis nach dem Komma abgeschnitten
		
		Fließkommazahlen: Ist mindestens ein Operand eine Fließkommazahl, wird das Ergebnis auf eine Fließkommazahl und nicht gerundet.
			int e = 3;
			float f = 4;
			float g;
			g = e / f;
				Binär

Rest (auch: Restwert Operator – Modulo genannt);
Berechnet den Rest der arithmetischen Division
	%	Ganze Zahlen, Fließkommazahlen	int k =11;
			int l = 5;
			int m;
			m = k % l;
				Binär

LOGISCHE OPERATOREN

Werten Ausdrücke zu wahr (true) oder falsch (false) aus

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel
Logische Komplement (Negation); Ändert der Wahrheitswert des Operanden	!	boolean	boolean b2 = false; boolean b1 = !b2;

Logisches UND;
Liefert true, wenn beide Operanden true sind.
	&&	boolean	boolean b4 = true;
			boolean b5 = true
			boolean b3 = b4 && b5;

Logische Oder;
Liefert true, wenn einer der beiden Operanden true ist
	||	boolean	boolean b7 = false;
			boolean b8 = true
			boolean b6 = b7 || b8;

Exclusiv-Oder;
Liefert true, wenn nur einer der beiden Operanden true ist
	^	boolean	boolean b10 = false;
			boolean b11 = true
			boolean b9 = b10 ^ b11;

VERGLEICHSOPERATOREN

Vergleichen Ausdrücke - liefern wahr (true) oder falsch (false)

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel

Gleichheit;
Primitive Datentypen:
Liefert true, wenn die Werte der Operanden gleich sind
	==	Primitive Datentypen	int z1, z2;
			boolean e1;
			z1 = 3;
			z2 = 3;
			e1 = z1 == z2;

Gleichheit;
Referenzdatentypen
Liefert true, wenn in beiden Opreanden die Refenrenz auf dasselbe Objekt enthalten ist
	==	Referenzdatentypen	Kunde kunde1, kunde2;
			boolean e2;
			kunde1 = new Kunde();
			kunde2 = kunde1;
			e2 = kunde1 == kunde2;

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel

Ungleichheit;
Primitive Datentypen:
Liefert true, wenn die Werte der Operanden nicht gleich sind
	!=	Primitive Datentypen	int z3, z4;
			boolean e3;
			z1 = 4;
			z2 = 3;
			e3 = z3 != z4;

Ungleichheit;
Referenzdatentypen
Liefert true, wenn in beiden Operanden nicht die Refenrenz auf dasselbe Objekt enthalten ist
	!=	Referenzdatentypen	Kunde kunde3, kunde4;
			boolean e4;
			kunde3 = new Kunde();
			kunde4 = new Kunde();
			e4 = kunde1 != kunde2;

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel

Kleiner als
liefert true, wenn der Wert des Linken Operanden kleiner ist verglichen mit dem Wert des rechten Operanden
	<	Ganz Zahlen, Fließkommazahlen	int z5, z6;
			boolean e5;
			z5 = 4;
			z6 = 5;
			e5 = z5 < z6;

Kleiner gleich
liefert true, wenn der Wert des Linken Operanden kleiner oder gleich ist verglichen mit dem Wert des rechten Operanden
	<=	Ganz Zahlen, Fließkommazahlen	int z5, z6;
			boolean e5;
			z5 = 4;
			z6 = 5;
			e5 = z5 <= z6;

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel

Größer als
liefert true, wenn der Wert des Linken Operanden größer ist verglichen mit dem Wert des rechten Operanden
	>	Ganz Zahlen, Fließkommazahlen	int z7, z8;
			boolean e6;
			z7 = 6;
			z8 = 5;
			e6 = z7 > z8;

Größer gleich
liefert true, wenn der Wert des Linken Operanden größer oder gleich ist verglichen mit dem Wert des rechten Operanden
	>=	Ganz Zahlen, Fließkommazahlen	int z7, z8;
			boolean e6;
			z7 = 6;
			z8 = 5;
			e6 = z7 >= z8;

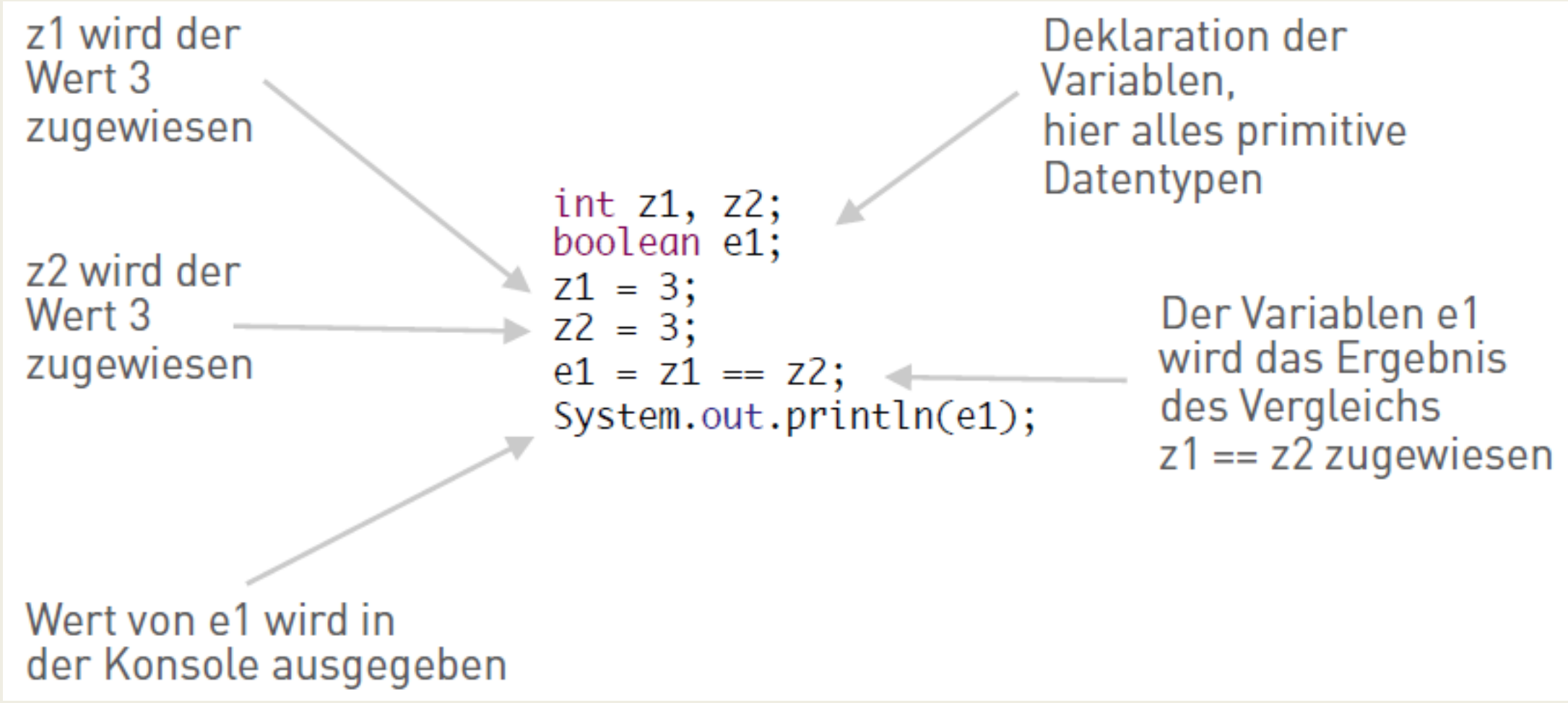
Operatoren und Ausdrücke

- Besondere Prüfoperatoren
  - *Prüfung auf Gleichheit (==)*
  - *Prüfung auf Ungleichheit (!=)*
- Unterschied beim Vergleich zwischen primitiven Datentypen und Referenzdatentypen
  - *bei primitiven Datentypen werden Werte verglichen*
  - *Bei Referenzdatentypen werden Referenzen verglichen*

VERGLEICH PRIMITIVER DATENTYPEN

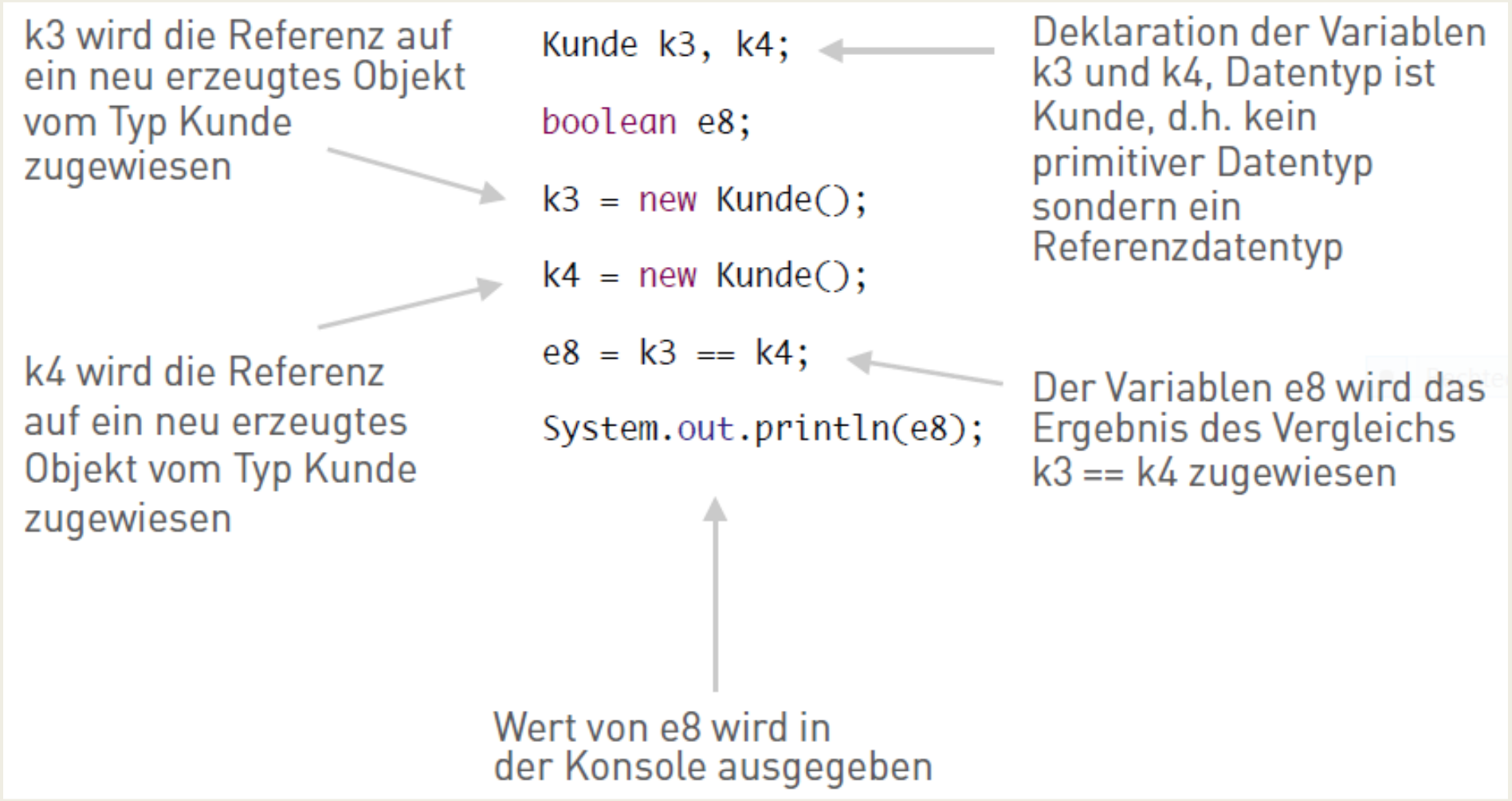
int, long, float, double, short, byte…

Operatoren und Ausdrücke



VERGLEICH VON REFERANZDATENTYPEN

Operatoren und Ausdrücke



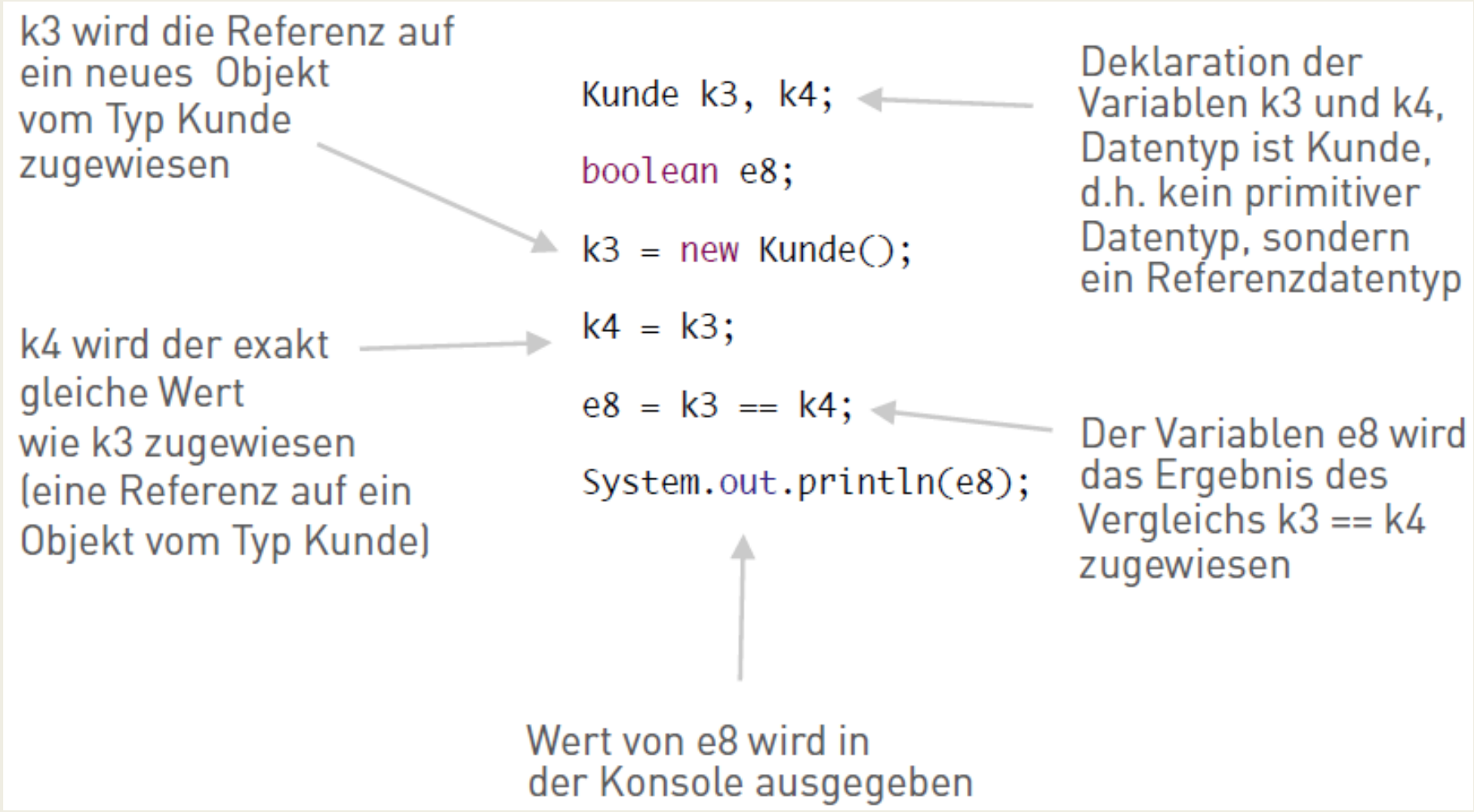
Operatoren und Ausdrücke

- k3 und k4 wird ein neu erzeugtes Objekt zugewiesen
  - *In k3 ist die Referenz auf das erste Objekt „Kunde“ gespeichert*
  - *in k4 die Referenz auf das zweite Objekt „Kunde“*
- Der Vergleich auf „Referenzgleichheit“ von k3 und k4 liefert damit den Wert false

Operatoren und Ausdrücke

- Codebeispiel wird angepasst
  - *k4 wird kein neu erzeugtes Objekt zugewiesen*
  - *k4 wird exakt der gleiche Wert zugewiesen, der in k3 gespeichert ist*
  - *k3 ist eine Referenz auf ein Objekt vom Typ „Kunde“*
  - *k4 hat nach der Anweisung k4 = k3 dieselbe Referenz als Wert wie k3*
- Damit liefert die Operation k3 == k4 das Ergebnis true.

Operatoren und Ausdrücke



OPERATOR ZUR VERBINDUNG VON ZEICHENKETTEN

Mehrere Zeichenketten zusammenführen

Konkatenation von mehreren Strings miteinander

Operatoren und Ausdrücke

Ausgeführte Operation	Operator	Anwendbar auf die Datentypen	Beispiel
Verketten von Zeichen (Konkatenation) Verketten von mehreren Zeichenketten zu einer neuen Zeichenkette	+	String	String s1, s2 s3 ,s4; s1 = „Hallo“; s2 = „ “; s3 = „Welt“; s4 = s1 + s2 + s3

PRIORITÄTEN

Bindungsstärke bzw. Vorrang der Operatoren

Operatoren und Ausdrücke

- **Priorität**: Bindungsstärke bzw. Vorrang der Operator
  - *Auswertungsreihenfolge folgt der Priorität*
  - *Regelt Vorrang bei unterschiedlichen Operatoren*

- Priorität der "Punkt-Operatoren" (*, /, %) höher als Priorität der "Strich-Operatoren" (+, -)
  - *d. h. Auswertung so, wie in der Mathematik üblich*

- Beispiel: Multiplikation vor Addition, also
  2 + 3 * 4 → 2 + 12 → 14

Operatoren und Ausdrücke

- Klammern (…) erzwingen Auswertungsreihenfolge
  - *gewünschte Auswertungsreihenfolge nicht immer gemäß Punkt vor Strich*
  - *Eingeklammerte Teilausdrücke immer zuerst ausgerechnet*
  - *Klammern sind um jeden Ausdruck erlaubt*
- Beispiele:
  - (2 + 3) * 4 → 5 * 4 → 20
  - *2 + (3 * 4) → 2 + 12 → 14*
  - *(2 + 3) → 5*
  - *(((2))) → 2*

Operatoren und Ausdrücke

- **Assoziativität** regelt Vorrang bei gleichrangigen Operatoren

- **Beispiel**: 8-3-2
  - *Linker Subtraktionsoperator zuerst: 8 - 3 - 2 → (8 - 3) - 2 → 5 - 2 → 3*
  - *Rechter Subtraktionsoperator zuerst: 8 - 3 - 2 → 8 - (3 - 2) → 8 - 1 → 7*

Operatoren und Ausdrücke

- Charakteristische Assoziativität eines Operators
  - *Links-assoziativ: am weitesten links stehender Operator wird zuerst ausgewertet*
  - *Rechts-assoziativ analog*
- Alle binären arithmetischen Operatoren sind linksassoziativ.
  - *Also 8 - 3 - 2 ⟶ (8 - 3) - 2 ⟶ 5 - 2 ⟶ 3*

Operatoren und Ausdrücke

Operator	Priorität	Assoziativität	Operanden	Bedeutung
+	1	rechts	1	Positives Vorzeichen
-	1	rechts	1	Negatives Vorzeichen
*	2	links	2	Multiplikation
/	2	links	2	Division
%	2	links	2	Modulo
+	3	links	2	Addition
-	3	links	2	Subtraktion