Java

Typen und Operatoren

CONTENTS

- Übersicht
- Primitive Typen
- Referenztypen
- Literale
- Operatoren
  - Allgemein
  - Mathematische
  - Weitere

PRIMITIVE TYPEN

Datentyp	Größe	Wertebereich	Definition	Wrapperklasse
boolean	1	false / true	Boolescher Wahrheitswert	java.lang.Boolean
char	2	U+0000, …, U+FFFF	Unicode	java.lang.Character
byte	1	-2^7, …, $+2^7-1$	8 bit signed Integer	java.lang.Byte
short	2	-2^{15}, …, $+2^{15}-1$	16 bit signed Integer	java.lang.Short
int	4	-2^{31}, …, $+2^{31}-1$	32 bit signed Integer	java.lang.Integer
long	8	-2^{63}, …, $+2^{63}-1$	64 bit signed Integer	java.lang.Long
float	4	$\pm 1{,}4e^{-45}$, …, $\pm 3{,}4e^{-38}$	32 bit IEEE 754 Floating Point	java.lang.Float
double	8	$\pm 4{,}9e^{-324}$, …, $\pm 1{,}7e^{-308}$	64 bit IEEE 754 Floating Point doppelter Genauigkeit	java.lang.Double
void	-	-	*Kein Wert*	*java.lang.Void*

STANDARDINITIALISIERUNG FÜR ATTRIBUTE VON PRIMITIVEM TYP

Primitiver Typ	Standard Initialisierung
boolean	false
char	\u0000 (null)
byte	(byte)0
short	(short)0
int	0
long	0L
float	0.0f
double	0.0d

REFERENZTYPEN

- Klassen
- Interfaces
- Beispiele aus Java Klassenbibliothek
  - Klasse String
  - Wrapper-Klassen für die primitive Typen: Integer, Long, Double, Float, … (mehr im Kapitel über Zahlen)
  - Interface AutoCloseable
  - Interface Comparable
  - Utilityklassen Collections, Arrays, Objects

LITERALE

Beispiel	Bedeutung	Bemerkung
1	Dezimal int	
42L	Dezimal long	
042	Oktal 34	Vorzeichen hängt vom
0xcafebabe	Hex int, -889275714	“High Order Bit” ab (später mehr dazu)
3.14F	float	
3.1415926535897 93D	double	
‘a’	char	
“abc”	String	
null, false, true	null Literal, boolesche Literale	

OPERATOREN

- Mathematische
- Relationale
- Logische
- Bitweise
- Zuweisungsoperatoren
- Weitere

MATHEMATISCHE OPERATOREN

Operator	Bezeichnung	Bedeutung
+	Positives Vorzeichen	+n ist wie n
-	Negatives Vorzeichen	-n dreht das Vorzeichen um
+	Summe	a + b
-	Differenz	a – b
*	Produkt	a * b
/	Quotient	a / b
%	Restwert	a % b gibt den Rest nach ganzzahliger Division (modulo)
++	Präinkrement	++a ergibt a+1 und erhöht a um 1
--	Prädekrement	--a ergibt a-1 und verringert a um 1
++	Postinkrement	a++ ergibt a und erhöht a um 1
--	Postdekrement	a -- ergibt a und verringert um 1

RELATIONALE OPERATOREN

Operator	Bezeichnung	Bedeutung
==	gleich	a==b ergibt true, wenn a gleich b ist.
!=	ungleich	a != b ergibt true, wenn a ungleich b ist
<	kleiner	a < b ergibt true, wenn a kleiner als b ist
<=	kleiner gleich	a <= b ergibt true, wenn a kleiner oder gleich b ist
>	größer	a > b
>=	größer gleich	a >= b

Achtung bei Fließkommazahlen!

LOGISCHE OPERATOREN

Operator	Bezeichnung	Bedeutung
!	Logisches NICHT	!a ergibt false, wenn a true ist
&&	Logisches UND mit Eval.abbruch	a && b ergibt true, wenn a und b true sind, ist a bereits false, wird b nicht bewertet
||	Logische ODER mit Eval.abbruch	a || b gibt true, wenn mindestens a ober b true sind. ist bereits a true, wird b nicht ausgewertet
&	Logisches UND	a & b ergibt true, wenn a und b true sind. a und b werden immer bewertet
|	Logisches ODER	a | b ergibt true, wenn mindestens a oder b true sind, a und b werden immer ausgewertet.
^	Logischer exklusiv oder	a ^ b ergibt true, wenn entweder a oder b true sind (genau einer von beiden)

BITWEISE OPERATOREN

Operator	Bezeichnung	Bedeutung
~	Einerkomplement	~a entsteht aus a, indem alle Bits in a invertiert werden
|	bitweises ODER	Wert von a | b: jedes Bit aus a und b jeweils per ODER verknüpft
&	bitweises UND	Wert von a & b: jedes Bit aus a und b jeweils per UND verknüpft
^	Bitweises Exklusiv ODER	Wert von a ^ b: jedes Bit aus a und b jeweils per Exklusiv ODER verknüpft
>>	Rechtsschieben mit Vorzeichen	a >> b : a wird um b Stellen nach rechts geschoben, das Vorzeichen wird beibehalten
>>>	Rechtsschieben ohne Vorzeichen	a >>> b: a wird um b Stellen nach rechts geschoben, oben wird mit 0 aufgefüllt
<<	Linksschieben	a << b: a wird um b Stellen nach links geschoben, unten wird mit 0 aufgefüllt

ZUWEISUNGOPERATOREN

Operator	Bezeichnung	Bedeutung
=	Einfache Zuweisung	a = b weist a den Wert von b zu
+=	Additionszuweisung	a += b : a = a + b
-=	Subtraktionszuweisung	a -= b : a = a - b
*=	Multiplikationszuw.	a *= b : a = a * b
/=	Divisionszuweisung	a /= b : a = a / b
%=	Modulozuweisung	a %= b : a = a % b
&=	UND-Zuweisung	a &=b : a = a & b
|=	ODER-Zuweisung	a | = b : a = a | b
^=	Exklusiv-ODER Zuw.	a ^= b : a = a ^ b
<<=	Linksschiebezuw.	a <<=b : a = a << b
>>=	Rechtsschiebezuw.	a >>= b : a = a >> b
>>>=	Rechtsschiebezuw. mit Nullexpansion	a >>>= b : a = a >>> b

WEITERE

Operator	Bezeichnung	Bedeutung
new		erstellt neues Objekt
==, !=	(nicht) identisch	Prüft Objektidentität
boolean ? st1 : st2		
Objekt::Methode	Methodenreferenz	
->	λ (lambda) Operator	arg -> code
(Typ) a	Type cast	Wandelt Typ von a in Typ um, wenn möglich
+	String Verkettung	Liefert einen neuen String aus den beiden Operanden