Variablen und Typen

- Was ist eine Variable?
- Wie wird sie in JAVA deklariert?
- Was ist ein Typ und wie wird er in JAVA realisiert?
- Einfache Datentypen in JAVA
- Wertzuweisung und Referenzzuweisung

Typ-Konzept

JAVA hat ein klares Typ-Konzept. Dabei ist ein *Typ* nichts anderes als eine Klasse. Das klare Konzept besteht darin, daß (fast) alles in JAVA einer Klasse zugeordnet ist: jedes Objekt, jede Variable, jede Konstante ist von einem Typ, d.h. gehört zu einer Klasse. Andersherum ausgedrückt: die Klasse gibt den Wertebereich der Variablen an.

Was ist eine Variable?

Variable Eine Variable ist ein Tripel (Name, Adresse, Wert). Der Name identifiziert die Variable. Die Adresse ist der Ort im Speicher, wo die Variable steht. Der Inhalt dieses Speicherplatzes ist der Wert der Variablen.

Konstante Eine Konstante ist eine Variable, deren Wert unveränderlich ist.

Typ in JAVA Der Wertebereich einer Variablen ist ihr Typ. Normalerweise sind alle Objekte einer Klasse **A** mögliche Werte – **A** muß angegeben werden.

Verwendung von Variablen in JAVA Variablen geben meist die Eigenschaften an, die Objekte einer Klasse haben.

Variablendeklaration

- Was für Werte kann die Variable annehmen? Welchen Typ hat sie?
- Wessen Eigenschaften beschreibt die Variable?
- Wie heißt die Variable? Wie kann sie von allen anderen Variablen unterschieden werden?

Intern wird dem Namen einer Variable eine Adresse zugeordnet. Dieser Speicherplatz, dessen Inhalt veränderlich ist, wird an den Namen der Variablen gebunden.

FieldDeclaration

$FieldDeclaration$:

$\quad Modifiers_{opt}\ Type\ VariableDeclarators$;

$VariableDeclarators$:

$\quad VariableDeclarator$

$\quad VariableDeclarators$, $VariableDeclarator$

$VariableDeclarator$:

$\quad VariableDeclaratorId$

$\quad VariableDeclaratorId = VariableInitializer$

$VariableDeclaratorId$:

$\quad Identifier$

$\quad VariableDeclaratorId$ `[ ]`

$VariableInitializer$:

$\quad Expression$

$\quad ArrayInitializer$

Modifikator

Sichtbarkeit:

private, *public* oder *protected*

Veränderbarkeit:

final oder das Fehlen des Modifikators *final*

Eigenschaft von Objekten oder Eigenschaft der Klasse: *static* Eine Klasseneigenschaft gibt es nur einmal, egal wieviele Objekte einer Klasse es gibt.

Eine Objekteigenschaft bekommt jedes Objekt der Klasse.

Einfache Datentypen

Unikate, nicht Objekte einer Klasse!

Typ	Inhalt	Standardwert	Größe
boolean	true, false	false	1 Bit
char	Unicode-Zeichen	u0000	16 Bit
byte	Integer mit Vorzeichen	0	8 Bit
short	Integer mit Vorzeichen	0	16 Bit
int	Integer mit Vorzeichen	0	32 Bit
long	Integer mit Vorzeichen	0	64 Bit
float	Fließkommazahl	0.0	32 Bit
double	Fließkommazahl	0.0	64 Bit

Wertzuweisungen

- Einfache Zuweisung:
  - $v = 5;$
    bedeutet, daß v den Wert 5 bekommt.
  - $s = “Zahn'';$
    bedeutet, daß s den Wert “Zahn” bekommt.
- Mehrfache Zuweisung:
  - $v = w = 5;$
    bedeutet, daß w den Wert 5 bekommt und dann v den Wert von w, also 5. Zuweisungen werden immer von rechts nach links durchgeführt.
  - $s = t = “Zahn'';$
    bedeutet, daß t als Wert “Zahn” bekommt und dann s den Wert von t, also “Zahn”.

Referenzzuweisung

Der Wert einer Variablen ist selbst wiederum eine Adresse, in der der eigentliche Wert steht. Soll eine Variable v den Wert einer anderen Variable w erhalten, so wird die Adresse, die bei w als Wert angegeben ist, kopiert und die Kopie der Adresse als Wert von v eingetragen.

Beispiel:

Variable mit Namen v, mit Adresse $a175$ und dort steht noch nichts.

w hat als Adresse $a100$.

Der Wert von w sei ein Objekt, z.B. Utas blauer Ball. Dies Objekt ist im Speicher unter der Adress $a200$ zu finden.

Im Speicherplatz $a100$ steht also "$a200$".

Nun soll v den Wert von w bekommen.

Name	Adresse	Wert
	vorher:	
v	$a175$	-
w	$a100$	$a200$
	nachher:	
v	$a175$	$a200$
w	$a100$	$a200$

Wertzuweisung direkt

Der Wert einer Variablen ist direkt unter der der Variablen zugeordneten Adresse eingetragen. Die Variable w übergibt direkt ihren eigentlichen Wert. JAVA verwendet die direkte Wertzuweisung bei Variablen, deren Wert von einfachem Datentyp ist.

Soll wieder die Variable v den Wert der Variablen w erhalten, wobei diesmal v und w vom einfachen Typ `double` sind, dann sieht die Wertzuweisung so aus:

Name	Adresse	Wert
vorher:		
v	$a175$	-
w	$a100$	0,324
nachher:		
v	$a175$	0,324
w	$a100$	0,324

Falls sich der Wert von w ändert, so bleibt der von v unverändert.

Operatoren

Für einfache Datentypen gibt es Operatoren. Alle anderen Aktionen müssen durch Methoden ausgeführt werden.

- Infixoperatoren:
  - $v = 2 + 3;$
  - $v = v + 3;$
    $v+ = 3;$
  - $s = \text{“}Zahn''$;
    $t = \text{“}rad''$;
    $u = s + t;$
- Präfixoperatoren:
  - $+ + v$ bzw. $- - v$
- Postfixoperatoren:
  - $v + +$ bzw. $v - -$
  - $w = v + +;$
    Jetzt hat w den Wert 4, v den Wert 5. Folgt nun
    $x = v;$
    so hat x den Wert 5.

Boole'sche Variablen – Bedingungen

- $==$ bedeutet die Gleichheit,

  sei z.B. $v = 2 + 3$ und $w = 5$, so ist bei

  $b = (v == w);$

  der Wert der Variable b vom Typ `boolean` `true`, also wahr.

- != bedeutet die Ungleichheit,

  so ist z.B.

  $b = (v! = w);$

  der Wert von b nun `false`, also unwahr, wenn v und w den Wert 5 haben.

- $>$ und $<$ bedeuten größer und kleiner.

  Sei b vom Typ `boolean` und $alter$ vom Typ `int`:

  $b = (alter > 18);$

- $>=$ und $<=$ bedeuten größer oder gleich bzw. kleiner oder gleich.

  $b = (v >= 18);$

Logische Operatoren

- $c \& d$ bedeutet das logische *und*
- $c|d$ bedeutet das logische *oder*
- $c\hat{}d$ bedeutet das *ausschließende oder*
- $!c$ ist wahr, wenn c falsch ist.

Wahrheitstafeln

a	! a
f	t
t	f

a	b	a & b
f	f	f
f	t	f
t	f	f
t	t	t

<table>
<tr><th>a</th><th>b</th><th>a | b</th></tr>
<tr><td>f</td><td>f</td><td>f</td></tr>
<tr><td>f</td><td>t</td><td>t</td></tr>
<tr><td>t</td><td>f</td><td>t</td></tr>
<tr><td>t</td><td>t</td><td>t</td></tr>
</table>

a	b	a^b
f	f	f
f	t	t
t	f	t
t	t	f

Beispiel für Bedingung und logischen Operator

```
boolean schulfrei;
int temperatur;
schulfrei = temperatur > 39 | 15 >= temperatur;
```

Die Bedingung wird zuerst ausgewertet, danach die logische Operation!