Java Datentypen

1. Grundbegriffe zur Datentypisierung

Unicode

Der UNICODE ist die Grundlage auf der ein Java Programm geschrieben wird, er ist praktisch ein Alphabeth, mit dem die Programme textlich formuliert werden.

Ein Java Programm besteht aus einer Vielzahl von Befehlen, die durch spezielle Programmanweisungen dem System mitgeteilt werden. Diese Anweisungen werden als Textkürzel geschrieben und sind im UNICODE kodiert. Der UNICODE stellt jedes Zeichen mit 16 Bit dar. Das heißt für die Darstellung von einem Zeichen werden 2 Bytes verwendet, was insgesamt die Darstellung von 65536 verschiedenen Zeichen ermöglicht. Der UNICODE ist also das erweiterte Alphabet um Daten und Informationen abzubilden. Im Gegensatz dazu steht der traditionelle ASCII Code, der ursprünglich 7 Bit breit war. Mit dem ASCII Code können Sie aber nur 128 verschiedene Zeichen abbilden, was für eine Darstellung sämtlicher weltweit verwendeter Schriftzeichen nicht ausreicht. Das Problem tritt z. B. bei den Email-Adressen auf, die auf dem ASCII Code aufbauen. Dort können Sie die deutschen Umlaute nicht verwenden, weil sie im ASCII Code nicht existieren. Die Verwendung des UNICODE spielt somit unter dem Aspekt des weltweiten Datenverkehrs eine entscheidende Rolle, da durch den UNICODE weltweit alle gängigen Schriftarten verwendet werden können.

Variablen

Zahlen und Textwerte werden in sog. Variablen im Arbeitsspeicher gehalten. Variablen sind benannte Speicherzellen und haben einen Datentyp. Der Datentyp sagt aus was in der Variablen abgelegt werden kann, also z. B. Zahlen oder Texte.

Unter Java wird eine Variable definiert durch:

<Datentyp> <Bezeichner>;

Es wird zunächst der Datentyp angegeben und danach der Name der Variablen.

Beispiel:

```
byte AnzahlderKinder;

double Faktor1;
```

Eine Variable kann initialisiert werden durch Angabe einer Wertzuweisung in der Definition:

```
byte AnzahlderKinder = 3;
```

Bezeichner

Die Bezeichner, also die Variablennamen, können vom Programmierer weitgehend frei gewählt werden. In Java können Bezeichner beliebig lang sein, wobei das erste Zeichen ein Buchstabe sein muss. Die restlichen Zeichen können dann, bis auf einige Steuerzeichen, frei gewählt werden. Beginnen muss ein Bezeichner mit einem Buchstaben. Zu den Buchstaben zählen hier auch das Dollarzeichen und der Bindestrich. Das Leerzeichen ist nicht erlaubt. Es wird zwischen Gross- und Kleinschreibung unterschieden.

Schlüsselwörter

Der Programmtext wird in Java mit speziellen Wörtern, den Schlüsselwörtern formuliert. Durch die Schlüsselwörter werden Datendefinitionen und Funktionalitäten beschrieben. In Java sind sämtliche Schlüsselwörter reserviert, d. h., dass die Schlüsselwörter nicht als Variablenbezeichner zulässig sind.

Tabelle 1: Schlüsselwörter unter Java

abstract	else	int	static
boolean	extends	interface	super
break	false	long	switch
byte	final	native	synchronized
byvalue	finally	new	this
case	float	null	throw
cast	for	operator	throws
catch	future	outer	transient
char	generic	package	true
class	goto	private	try
const	if	protected	var
continue	implements	public	void
default	import	rest	volatile
do	inner	return	while
double	instanceof	short	

2. Datentypen

Jede Variable, die wir anlegen basiert auf einem Datentyp. Er bestimmt was in dieser Variablen prinzipiell abgelegt werden kann. Wir können Variablen auch mit Schubkästen vergleichen, in die wir Werte reinlegen können. Denken wir uns mal Schubfächer in die nur Zahlen reinpassen und Schubkästen in die nur Buchstaben reingehen, dann haben wir bereits die zwei grundlegenden Datentypen „nummerisch“ und „alfanummerisch“ eingeführt. In Java sind die fundamentalen Ganzzahlen-Datentypen, die Gleitkomma-Datentypen, der Zeichen- Datentyp und der Boolsche Datentyp realisiert, sie zählen zu den Primitiven Datentypen. Referenz-Datentypen umfassen Variablen von Klassen, Feldern und Strings.

Ganzahlentypen

Die Gruppe der Ganzzahlen-Typen wird in byte, short, int und long unterteilt. Diese vier Typen haben alle gemein, dass nur ganze Zahlen in ihnen abgelegt werden können, also keine Texte und keine Zahlen mit Nachkommastellen. Eine Variable vom Typ byte kann Werte so von –128 bis +127 tragen.

Tabelle 2: Wertebereich von Ganzzahlentypen

Type	Speicherung	Wertebereich
byte	8 Bit	-128 bis 127
short	16 Bit	-32768 bis 32767
int	32 Bit	- 231 bis 231 - 1
long	64 Bit	- 263 bis 263 - 1

Gleitkomma Typen

Neben diesen Ganzahlentypen kennen wir in Java noch die Kategorie der Gleitkommazahlen. Hier existiert der Typ float und der Typ double. Mit diesen Typen können Variablen definiert werden, die über Nachkommastellen und einen Exponenten verfügen. Dabei kann eine Variable vom Typ float von –3,4 1038 bis +3,4 1038 und eine Variable vom Typ double von ca. -1,8 10308 bis +1,8 10308 reichen. Variablen vom Typ float haben eine Genauigkeit von 8 und Variablen von Typ double eine Genauigkeit von 16 Stellen nach dem Komma.

Tabelle 3: Wertebereich von Gleitkommatypen

Typ	Speicherung	Wertebereich
float	32 Bit	-3,40282347 1038 bis 3,40282347 1038
double	64 Bit	- 1,797693138462315750 10308 bis 1,797693138462315750 10308

Boolean

Um reine zweiwertige Aussagen zu verarbeiten verwendet man den Typ Boolean. Variablen von diesem Typ können nur die Werte true oder false annehmen, also keine Zwischenwerte. Sie können eine boolsche Variable z.B. dann verwenden wenn Sie die Daten von Artikeln speichern wollen und dort zwischen Eigenfertigung und Fremdfertigung unterscheiden. Der Typ boolean wird weiterhin gerne eingesetzt um Ergebnisse logischer Vergleiche, also z.B. A>0, abzuspeichern und diese auszuwerten.

Char

Eine Variable vom Typ char kann genau ein Zeichen speichern. Der char-Typ verwendet zur Darstellung wiederum den UNICODE, so dass für ein Zeichen 16 Bit benötigt werden. Damit können 65536 verschiedene Zeichen dargestellt werden. Auf der Basis des Typs char können weitere Texttypen, wie z. B. Zeichenketten, die sog. strings, gebildet werden.

Casts

An dieser Stelle soll auch noch den Begriff der Cast-Operationen eingehen. Unter Cast versteht man eine Typanpassung bei einer Wertzuweisung. Angenommen Sie Sie wollen einer Variablen „Anzahl der Kinder“, die vom Typ byte ist, den Inhalt einer Variablen zuweisen, die vom Typ float ist. Die Zielvariable also „Anzahl der Kinder“ ist weniger mächtig so dass Teile der Zahl u. U. abgeschnitten werden. Eine solche Typanpassung kann implizit oder explizit erfolgen. Impliziete Anpassungen sind immer dann möglich, wenn der Zieltyp der Variablen eine höhere Genauigkeit aufweist, also z.B. Sie weisen einer Variable vom Typ float eine Variable vom Typ int zu. Das geht ohne Problem und ohne irgend eine zusätzliche Angabe. Wollen Sie aber einer ungenaueren Variablen eine genauere zuweisen, müssen Sie eine expliziete Typanpassung durchführen. Dies erfolgt indem Sie den Zieldatentyp, in den umgewandelt werden soll in Klammern vor die Variable schreiben. Aber hier ist Vorsicht geboten, da mit dieser Anpassung möglicherweise Informationen abgeschnitten werden.

Felder (eng. arrays)

Datenfelder sind Datentypen, bei denen in einer Variablen eine Vielzahl von Werten Platz findet. Man könnte es damit vergleichen, dass sich in einer aufgezogenen Schublade nochmal ein kleiner Kasten mit durchnummerierten Schubladen befindet. Anhand der Nummer auf der Schublade kann auf den Inhalt des Feldes zugegriffen werden. Die Nummer auf der Schublade wird hier auch als Index bezeichnet.

Beispiel:

```
int [] Jahr = new int [9]


Jahr[0] = 1996;


Jahr[2] = 1910;


Jahr[4] = 2000;
```

Durch die Anweisung int eckige Klammer auf, eckige Klammer zu wird ein Feld angelegt. Es hat den Namen Jahr und kann neun Integerwerte aufnehmen. In diesem Feld sollen z. B. die Baujahre von Immobilien gespeichert werden. Die Identifikation erfolgt über den Index. Hier wird z.B. dem ersten Wert (Index 0) die Jahreszahl 1996, dem dritten Wert (Index 2) die 1910 und dem fünften Wert(Index 4) die 2000 zugewiesen. Der Eintrag mit dem Index 1 und 3 bleibt hier leer. Wie man sieht können in der Variablen Jahr, gleichzeitig mehrere gleichartige Daten abgelegt werden können.

3. Klassen

In Java existieren von Haus aus bewusst nur die bisher dargestellten einfachen Typen. Eine zentrale Stärke der Programmiersprache Java ist aber die objektorientierte Programmierung. D. h., dass der Programmierer seine Datentypen selbst definieren und damit sehr kreativ werden kann. Eine Klasse beinhaltet zunächst Variablen in denen Werte gespeichert werden und zusätzlich Methoden, die auf diese Werte angewendet werden. In den Methoden werden die Funktionalitäten, die auf den neuen Datentyp möglich sein sollen programmiert. Die Methoden sind mit den Operatoren auf elementaren Datentypen zu vergleichen. Wie Sie wissen

können Sie auf Zahlen eine Addition durchführen, so dass der + Operator bei Zahlentypen der Methode Addition entspricht.

Klassendefinition

Die Definition einer Klasse erfolgt mit dem Schlüsselwort class. In der Definition werden zunächst Variablen definiert und danach die Methoden auf die Datenstruktur.

Beispiel Klasse: Datum

```
class datum;
{
byte tag;
byte monat;
int jahr;

public static void naechstertag();

public static string monatsname();

}
```

Hier im Beispiel wird eine Klasse definiert, die den Datentyp datum repräsentiert. Die Klasse wird eingeleitet durch das Schlüsselwort class – es folgt der Name der Klasse, also Datum. In den geschweiften Klammern werden nun einerseits Daten definiert, hier der tag, der monat und das jahr. An diesem Beispiel sieht man auch deutlich, dass ein neuer Datentyp unterschiedliche Datentypen enthalten kann. Nach der Deklaration der eigentlichen Daten erfolgt die Formulierung der Methoden, also der Operationen auf den neuen Datentyp datum. Auf datum seien hier nur zwei Methoden angelegt. Die erste Methode naechstertag würde den Datumswert um einen Tag weiterschalten – in der zweiten Methode monatsname würde der Monatsname als Text ermittelt und an das aufrufende Objekt übergeben. Wichtig ist, dass der neue Datentyp datum einerseits die Datenkomponenten enthält und andererseits die Methoden, also die Funktionalitäten.

Wie im Beispiel zu sehen ist, kann eine Methode verschiedene Eigenschaften erhalten. Hier sehen wir static, public und void. static bedeutet, dass die Methode statisch, also nur ein einziges Mal in der Klasse existiert, auch wenn mehrere Objekte von dieser Klasse generiert werden. public bedeutet, dass die Methode auch von außerhalb der Klasse aufgerufen werden darf. Das Gegenteil wären private Klassen die nur innerhalb der Klasse aufgerufen werden können. void bedeutet, dass die Klasse dem aufrufenden Programm keinen Wert zurückgibt. Wir sehen das hier am Beispiel von naechstertag. Wird die Methode naechstertag aufgerufen, wird in der Klasse auf den nächsten Tage geschaltet aber dem Aufrufer kein Wert zurückgeliefert. Anders ist das bei monatsname, hier wird die textliche Monatsbezeichnung als string, also als Zeichenkette, zurückgegeben.

Ausblick auf Klassenbibliotheken

In der Grundversion verfügt Java nur über die wichtigsten Funktionen. Die Hersteller von Java Compiler liefern aber eine Vielzahl von Klassenbibliotheken mit, durch die dann wiederum komplexe Ein/Ausgabe-Funktionen, Grafische Benutzeroberflächen und so weiter realisiert werden

Übungen

Welche der folgenden Variablennamen sind ungültig?

```
EineZahl

Zähler

Rechnungs_Nr

Millionen$

Zahl_Nr.1

10erStelle

Egalité

ja/nein

ganz_ganz_langer_VariablenName

Typ.1

_$

Asterix&Obelix
```