

# Programmieren I

Dr. Klaus Höppner

Hochschule Darmstadt – Wintersemester 2009/2010

Grundlagen von JAVA

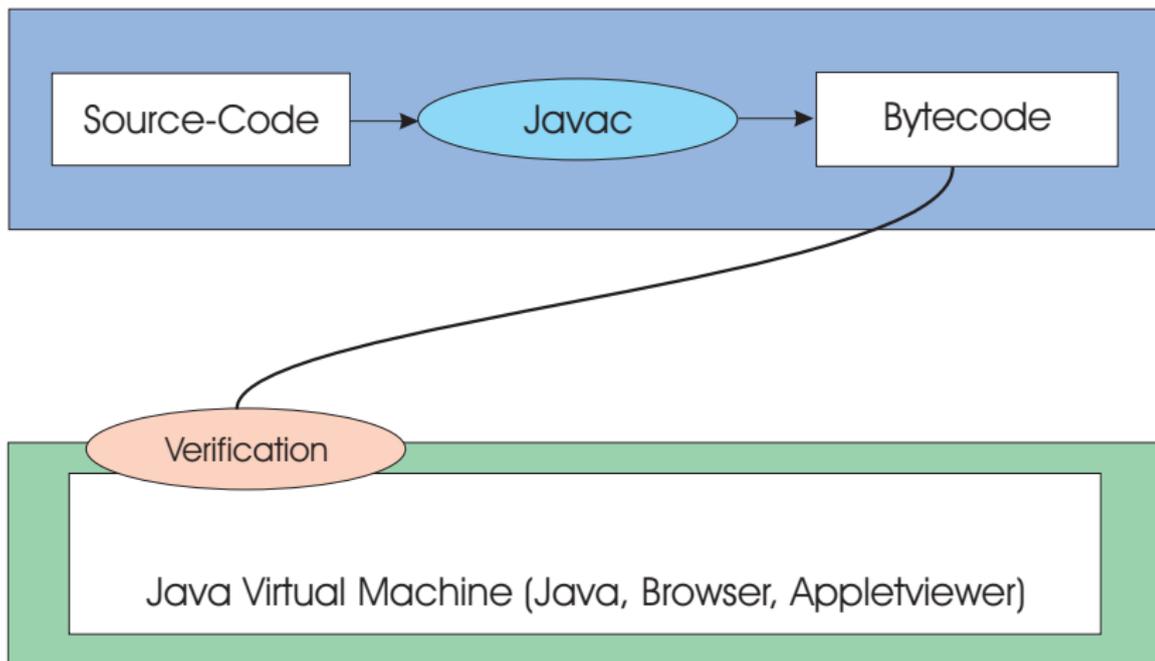
Hello World in JAVA

Elementare Datentypen

## Eigenschaften von JAVA

- Plattform-und Betriebssystem-unabhängig
- Einfach zu nutzende Programmiersprache
- Objekt-orientiert
- interpretierbar, aber dennoch schnell
- eigenes Speichermanagement
- Multi-Threaded
- Das ist C++ außer dem „Hard Stuff“:
  - kein programmiertes Speichermanagement
  - keine expliziten Pointer
  - kein Operatorenüberladen
  - keine Strukturen oder Unions
  - keine Mehrfachvererbung
- „The Key To Transparent, Universal Connectivity And Information Exchange“

## Prinzipieller Ablauf



## Geschichtliches

- Anfang der 1990er Jahre: Programmiersprache Oak
- aus rechtlichen Gründen in JAVA umbenannt (Café „Java City“, das die Entwickler in den Pausen besuchten)
- Auftraggeber Sun, anfänglich aber ansonsten wenig beteiligt
- 1995/6: Version 1, bereit mit wesentlichen Standardbibliotheken:
  - `java.lang` elementare Klassen, z. B. Object und System
  - `java.io` Ein- und Ausgabe
  - `java.util` z. B. Datenstrukturen und eine Datumsklasse
  - `java.net` einfache TCP/IP-Klassen
  - `java.awt` grundlegende Klassen für grafische Benutzeroberflächen
  - `java.applet` eine Klasse für Applets

## Geschichtliches (Forts.)

- 1997: Version 1.1
  - Internationalisierung
  - Java-Komponenten: *Java Beans*
  - Dateiformat JAR für die Paketierung ganzer Bibliotheken oder Anwendungen
  - Verteilte Programme: Remote Method Invocation (RMI)
  - JDBC: Zugriff auf relationale Datenbanken
  - Sicherheitsfunktionen
- 1998: Version 1.2 („Java 2“)
  - Verbesserung der Performance
  - neue Schnittstellen und Implementierungen für Collections
  - die Oberflächen-Bibliothek Swing
  - eine CORBA-Schnittstelle

## Geschichtliches (Forts.)

- 2000: Version 1.3
  - Bugfixes
- 2002: Version 1.4
  - Serverprogrammierung (z. B. Servlets)
- 2004: Version 5.0 (=1.5)
  - Generische Typen (Generics)
  - Vereinfachte Syntax für die Iteration über Collections, Maps und Arrays
  - Aufzählungswerte („Enumerations“)
- 2006: Version 6.0 (=1.6)
  - Web 2.0

# Erstes JAVA-Programm

*// Erstes Programm*

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hallo Welt");  
    }  
}
```

# Analyse des Beispiels

- Die Zeile

*//Erstes Programm*

ist ein Kommentar

- JAVA kennt zwei Arten von Kommentaren:
  1. Einzeilige Kommentare: Eingeleitet mit *//*  
Der Kommentar gilt jeweils vom *//* bis zum Ende der Zeile.
  2. Kommentar-Bereich:  
Alles zwischen */\** und *\*/* gilt als Kommentar.  
Dieser Bereich kann auch über mehrere Zeilen gehen.

## Analyse des Beispiels (Fortsetzung)

- Jedes JAVA-Programm besteht aus mindestens einer Klasse:  
`public class HelloWorld { ... }`
- Die Datei mit dem Quelltext muss dabei so heißen wie die Klasse, mit der Endung „.java“.
- Das Hauptprogramm befindet sich in einer statischen Methode (`public static void main(...)`)
- Blöcke von Anweisungen werden in geschweiften Klammern (`{...}`) eingeschlossen.
- Anweisungen werden mit einem Semikolon abgeschlossen.
- Die Formatierung des Quelltextes ist unwichtig.

# Variablen

In JAVA können Variablen definiert werden. Diese sind entweder Objekte oder von einem elementaren Datentyp.

Variablen können sein:

- lokale Variablen innerhalb einer Methode
- Member-Variablen (normale Attribute)
- Klassen-Variablen (statische Attribute)

## Variation des Beispiels

```
public class HelloWorld {  
    public static void main(String[] args) {  
        BufferedReader in =  
            new BufferedReader(new InputStreamReader(System.in));  
  
        System.out.print("Ihr Name: ");  
        try {  
            name = in.readLine();  
            System.out.println("Hallo "+name);  
        } catch (IOException e) {  
        }  
  
    }  
  
    static String name;  
}
```

# Übersicht elementarer Datentypen

Typ	Bytes	Wertebereich
byte	1	-128...127
short	2	-32768...32767
int	4	-2147483648...2147483647
long	8	$\pm 9,2 \times 10^{18}$
float	4	Gleitkommazahl (einfache Genauigkeit)
double	8	Gleitkommazahl (doppelte Genauigkeit)

## Kleiner Exkurs: Zahlensysteme

Das bei Menschen übliche Zahlensystem ist das Dezimalsystem (Basis 10).

Es gilt also:

$$127_{10} = 1 \times 10^2 + 2 \times 10^1 + 7 \times 10^0$$

Im EDV-Bereich sind noch das Binär- und Hexadezimalsystem von Bedeutung. Das Hexadezimalsystem kennt die Ziffern 0-15, nämlich 0...9, A...F.

$$1010_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 10_{10}$$

$$A7_{16} = 10 \times 16^1 + 7 \times 16^0 = 167_{10}$$

- Computer speichern Daten im Binärsystem.
- Kleinste Einheit ist das Bit (*binary digit*): 0 oder 1.
- Eine Kette von 8 Bit heißt Byte. Diese kann  $2^8 = 256$  verschiedene Werte annehmen: 0–255.
- Jedes Byte lässt sich durch zwei Hexadezimal-Ziffern darstellen:

$$\underbrace{1011}_B \underbrace{0101}_5 {}_2 = B5_{16} = 181_{10}$$

- Gebräuchliche Einheiten:

$$1 \text{ KB} = 1024 \text{ B} \quad (= 2^{10} \text{ B})$$

$$1 \text{ MB} = 1024 \text{ KB} \quad (= 1048576 \text{ B})$$

$$1 \text{ GB} = 1024 \text{ MB} \quad (= 1073741824 \text{ B})$$

## Negative Zahlen

Ein Byte kann 256 verschiedene Werte, nämlich  $00000000_2$  bis  $11111111_2$  annehmen. Dies kann man als Werte 0 bis 255 interpretieren.

Wie stellt man negative Zahlen dar? *Per Definition* kann man das höchstwertige Bit eines Datentyps als Vorzeichen-Bit interpretieren. Bei einem Datentyp, der aus einem Byte besteht, gelten in diesem Fall die Werte

$$00000000_2 \dots 01111111_2 = 0_{10} \dots 127_{10}$$

als positiv. Die Werte

$$10000000_2 \quad (= -128_{10}) \dots 11111111_2 \quad (= -1_{10})$$

gelten als negativ.

# Beispiel

```
import java.io.*;

public class HelloWorld {
    public static void main(String[] args) {
        BufferedReader in =
            new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Ihr Alter: ");
        try {
            alter = Integer.parseInt(in.readLine());
            System.out.println("Jahre bis zur Rente: "
                +String.valueOf(rentenalter-alter));
        } catch (IOException e) {}

    }

    static int alter;
    final static int rentenalter=67;
}
```

# Literale Integer-Werte

Ganze Zahlen können auf drei Arten dargestellt werden:

1. Dezimal:

```
int i = 127;
```

2. Hexadezimal:

Hexadezimalwerte beginnen mit 0x oder 0X.

```
int i = 0x28; // i = 40 (dezimal)
```

## Besondere Schlüsselworte

- Bei Variablen, deren Wert sich im Programm nicht mehr ändert, kann man bei der Deklaration das Schlüsselwort `final` benutzen. Diese müssen dann schon bei der Deklaration mit einem Wert versehen werden:

```
final double pi = 3.1415927;
```

- Eine Deklaration von Variablen als `final` schützt vor versehentlichem Ändern.

## Felder – Arrays

- Ein Array ist ein Feld (bzw. eine Folge) von Variablen gleichen Typs.
- Bei der Deklaration wird die Länge der Folge in eckigen Klammern angegeben:

```
int i[] = new int[10];  
i[3] = 17;  
System.out.println(i[3]);
```

- Der Index eines Array startet immer bei 0, d. h. der Index läuft von  $0 \dots n - 1$
- Benutzt man einen Index, der außerhalb der Grenzen des Arrays liegt, so wird eine Exception geworfen.

⋮		Index	
17	0	←	Tabelle
35	1		
112	2		
-3	3		
1000	4		
⋮			

```
int tabelle[] = new int[5];  
tabelle[0]=17; tabelle[1]=35; ...
```