Einführung in die Programmierung mit Java

Martin Wirsing

in Zusammenarbeit mit
Michael Barth und Gefei Zhang

Ziele

- Begriffsbildungen: Informatik, Algorithmus, Programm ...
- Warum Java als Programmiersprache verwenden?
- Ein einfaches Java-Programm erstellen, übersetzen und ausführen
- Gut dokumentierte Java-Programme erstellen können

Informatik

Informatik

ist ein Kunstwort,
das in den 60-ziger Jahren in Frankreich kreiert wurde,

entstanden aus
Information + Mathematik

englisch: Computer Science
neuerdings auch: Informatics

bedeutet
Wissenschaft der maschinengestützten Informationsverarbeitung

Teilgebiete der Informatik

Praktische Informatik

- Programmierung und Software-Entwicklung
- Datenbanksysteme
- Betriebssysteme und Middleware

Technische Informatik

- Rechenanlagen und Rechnernetze (Hardware)

Theoretische Informatik

- Formale Modelle und Semantik
- Algorithmen und Komplexität

Informatik und Gesellschaft

- Auswirkungen der Informatik auf die Gesellschaft
  (Rationalisierung, Automatisierung, Datensicherheit, ...)

Algorithmen und Programme

Algorithmus (nach Al-Khowarizmi, um 800)

- Allgemeines Verfahren zur Lösung einer Klasse von Problemen, das durch eine eindeutige Vorschrift so genau festgelegt ist, dass man es anwenden kann, ohne es verstanden zu haben.
- Eigenschaften:
  - Jeder Einzelschritt ist eindeutig festgelegt und berechenbar.
  - Das Verfahren liefert nach endlich vielen Schritten eine Lösung.
- Beispiele:
  - Modellbau: Montageanleitung
  - Informatik: Sortieralgorithmus

Programm

- Beschreibung von Datenstrukturen und Algorithmen in einer Programmiersprache

Programme und Software-Entwicklung

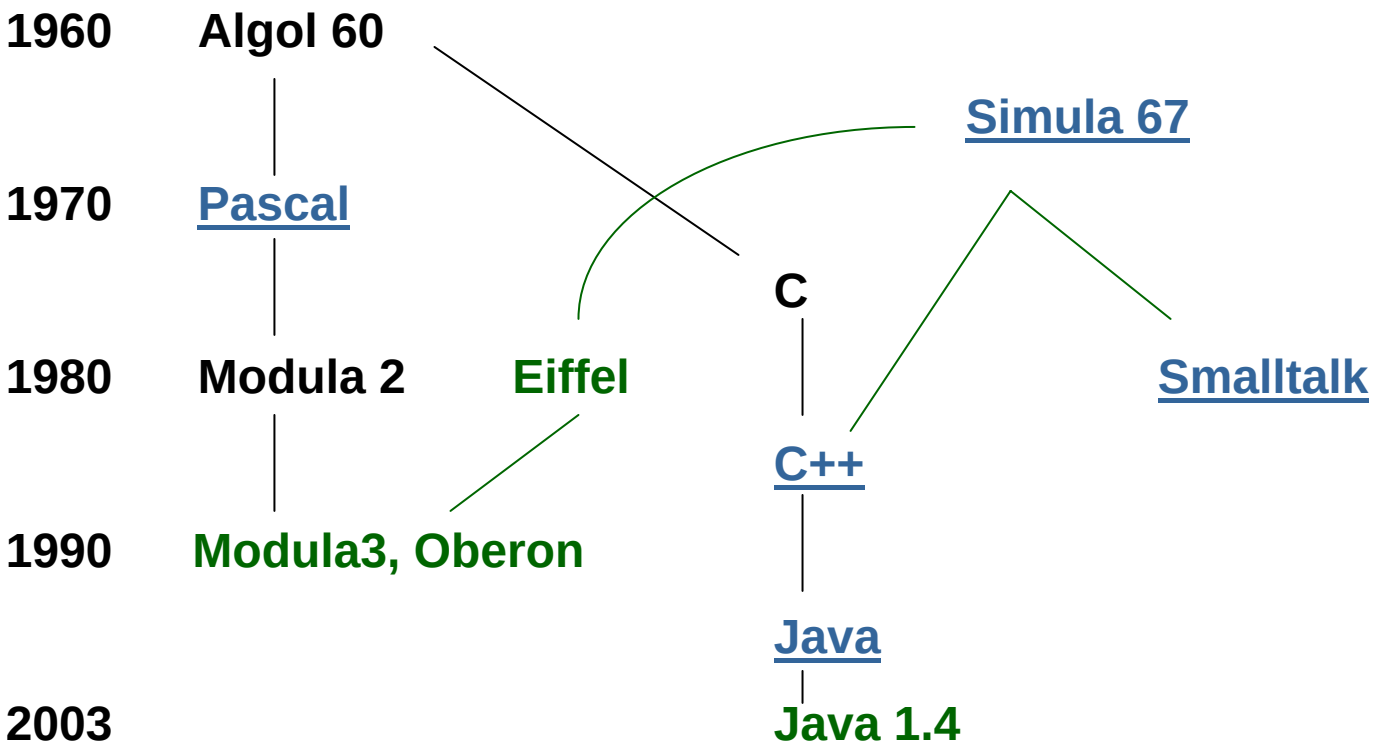
Programm

- Beschreibung von Datenstrukturen und Algorithmen in einer Programmiersprache

Software-Entwicklung

- Systematische Konstruktion von Programmen zur Lösung eines in der realen Welt gestellten Problems

Entwicklung objektorientierter Programmiersprachen



Java

- Entwickelt von J. Gosling, u.a.
- Erste plattform-unabhängige OO-Sprache, insbesondere zur Programmierung von Internet-Applikationen
- Erste Version 1.0 1995, heute Java 1.4. (neu jetzt 1.5. aber nicht in Vorlesung)
- Ursprünglicher Name: OAK.

Aspekte von Java

- Objektorientiert: Klassenkonzept, strenge Typisierung
- Unabhängig von Plattform: Durch Übersetzung in Virtuelle Maschine (JVM)
- Netzwerkfähig, nebenläufig
- Sicherheitskonzept

- **Nachteile:** Laufzeithandicap durch Interpretation der JVM

  (aber z.T. ausgeglichen durch Just-in-Time Übersetzung)
- **Vorteile:**
  - Verteilte Anwendungen, Web-Anwendungen
  - Rechnerunabhängigkeit von Graphikanwendungen

Grober Aufbau eines Java-Programms

- **Java-Programm** besteht aus Menge von **Klassen**

- Eine **Klasse** besteht aus
  - **Attributen** („fields“): Beschreiben **Zustand eines Objekts**
  - **Methoden**: Beschreiben die **Operationen**, die ein Objekt ausführen kann

Einfaches imperatives Java-Programm

- Ein **imperatives** Java-Programm besteht aus
  - **Klassendeklaration** mit einer
  - **einzigen Methode** namens **“main”:**

```
public class <KlassenName>
{
        public static void main(String[] args)
        {
                <Anweisungen>
        }
}
```

Beispiel: Hallo

```
public class Hallo
{
        public static void main(String[] args)
        {
                System.out.println(„Hallo!“);
        }
}
```

Methodenaufruf

- **Methodenaufruf allgemein:**

  object.methodName(parameters);

- **Beispiel:**

```
System.out.println(„Hallo!“);
```

Konventionen

- Klassennamen beginnen mit großen Buchstaben

  Bsp. Hallo

- Methodennamen und Variablennamen beginnen mit kleinen Buchstaben

  Bsp. println, out

- Konstantennamen bestehen NUR aus großen Buchstaben.

  Bsp. BLUE

- Zusammengesetzte Namen werden zusammengeschrieben,
  jeder (innere) Teilname beginnt mit einem großen Buchstaben

  Bsp. Klasse HalloWelt, Methoden getName, getMyObject

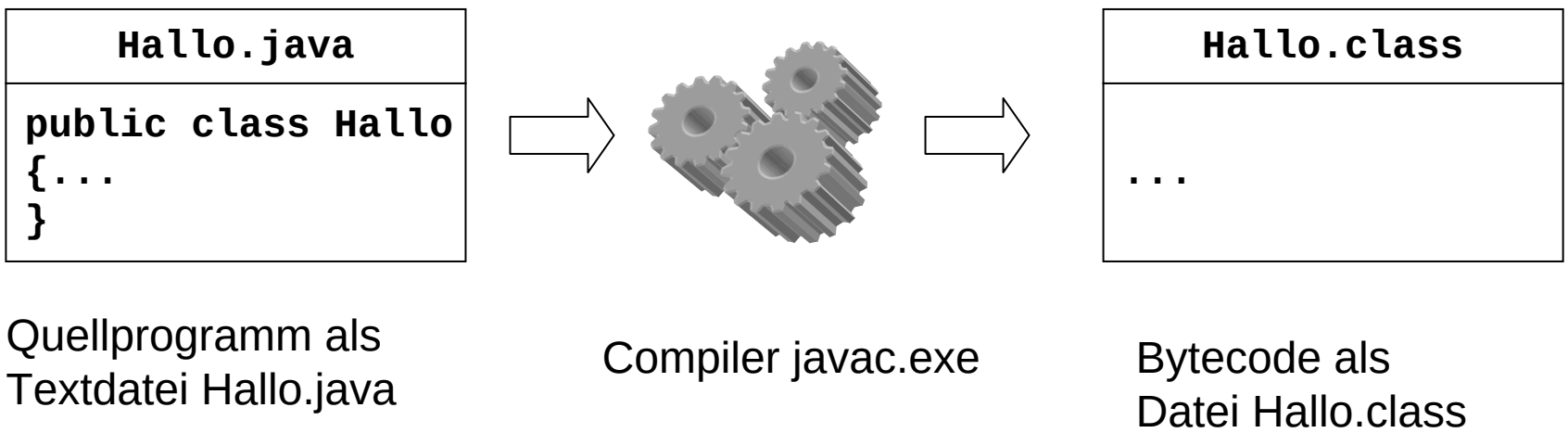
Übersetzung und Ausführung von Java-Programmen

Übersetzung in Bytecode

- Aus einer Textdatei mit Endung „.java“ erzeugt der Compiler javac eine

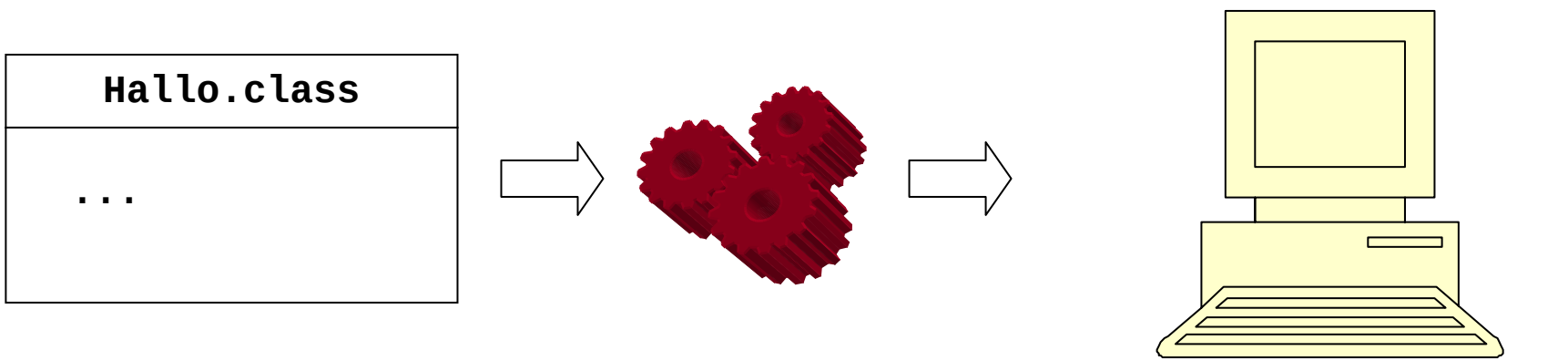
Datei mit gleichem Namen, aber Endung „.class“

- Diese enthält den Bytecode für die JVM



Übersetzung und Ausführung von Java-Programmen

- Die Datei mit dem Bytecode wird der JVM übergeben und von der JVM ausgeführt (d.h. interpretiert).



Übersetzung und Ausführung von Hallo.java (unter Windows)

Übersetzung von Hallo.java:

```
C: > javac Hallo.java
```

Interpretation von Hallo.class:

```
C: > java Hallo
```

Gibt auf Bildschirm zurück:

```
Hallo!
```

Kommentare in Java

„The view that documentation is something that is added to a program after it has been commissioned seems to be

wrong in principle, and counterproductive in practice.

Instead, documentation must be regarded as an integral part of the process of design and coding. „

C. A. R. Hoare:

Hints on Programming Language Design (1973)

Darstellungen für Kommentare in Java

- Durch

  ```
  // bla, bla}
  ```

  wird eine Zeile oder ein Rest einer Zeile zum Kommentar.

- Zur Erzeugung von Kommentaren zu Klassen und Methoden werden die Klammern

  `/**` und `*/`

  verwendet.

  Solche Kommentare werden in den mit dem Befehl **`javadoc`** erzeugten Report mit aufgenommen.

Die Klasse Hallo dokumentiert

```
 /**
Diese Klasse dient nur zum Anzeigen des Strings "Hallo,
Welt!„ auf den Bildschirm
 */
public class HalloDoc
{       /**    Die Methode main druckt "Hallo, Welt!„
         */
        public static void main (String[] args)
        {       System.out.println("Hallo, Welt!");
        }
}
```

Erzeugung der Dokumentation

- Mit dem Befehl

```
javadoc HalloDoc.java
```

wird automatisch eine Beschreibung der Klasse `HalloDoc` erzeugt und in die Datei

```
HalloDoc.html
```

geschrieben.

Spezielle Variablen bei javadoc

- `@see` für Verweise
- `@author` für Namen des Autors
- `@version` für die Version
- `@param` für die Methodenparameter

Die Klasse Square ausführlich dokumentiert

```
 /** Diese Klasse dient zur Berechnung des Quadrats.
@author Martin Wirsing
@version 1.1
 */
public class Square
{       /**  Diese Methode dient nur zur Illustration der
        Parameterbehandlung durch javadoc.
        @param value ist ein formaler Parameter vom Typ int
        @return das Quadrat von value
         */
        public static int square (int value)
        { return value*value; }
}
```

Eine Testklasse für Square

```
 /**    Diese Klasse dient nur zum Test von Square
 */
public class Programm
{       /** Die Methode main druckt einen Testfall von square
         */
        public static void main (String[] args)
        {       int wert = 17;
                System.out.println("Das Quadrat von "
                        + wert + " ist " + square(wert));
        }
}
```

Erzeugung der Dokumentation

- Mit dem Befehlen

```
javadoc Square.java
javadoc Programm.java
```

werden automatisch Beschreibungen der Klassen `Programm` und `Square` erzeugt und in die Dateien

```
Square.html und Programm.html
```

geschrieben.

Zusammenfassung

- **Geschichte:**
  - Objektorientierte Programmiersprachen seit 1967: Simula
  - OO-Programmierung populär seit Ende der 80er Jahre mit Smalltalk und C++.
  - Heute vor allem C++ und Java; neuerdings auch C#
- **Java**
  - OO-Programmiersprache,
  - vor allem zur Programmierung im Internet eingesetzt
  - Java ist plattform-unabhängig, interpretierend, unterstützt Sicherheitskonzepte und besitzt eine reichhaltige Klassenbibliothek (API, engl. "Application Programming Interface").

Zusammenfassung (2)

- Ein **Java-Programm** besteht aus einer oder mehreren **Klassen**.
  - Klassen enthalten **Attribute** und die Definitionen von **Methoden**.
  - Eine Methode besteht aus einer **Sequenz von Anweisungen**, die den Berechnungsablauf festlegen.
- Jede **selbstablaufende** Java-Anwendung enthält eine Methode „**main**“.
- Ein Java-Programm wird mit einem **Übersetzer in Byte-Code** übersetzt, der dann mit einem **Interpreter**, der **JVM** , ausgeführt wird.
- Java-Programme sollten gut dokumentiert werden.
  Mit **`javadoc`** kann automatisch eine übersichtliche Dokumentation erzeugt werden.

Literaturhinweise

- K. Arnold, J. Gosling. *The Java Programming Language.* Addison-Wesley, 1996.
- C. Horstmann. *Computing Concepts with Java Essentials.* 3rd Edition, Wiley, 2003.
- H.-P. Gumm, M. Sommer. *Einführung in die Informatik.* 6. Auflage, Oldenbourg-Verlag, 2004.

Online-Material:

- G. Krüger. *Handbuch der Java-Programmierung.* 4. Auflage, Addison-Wesley, 2004.
- C. Ullenboom. *Java ist auch eine Insel.* 4. akt. und erw. Auflage Galileo Computing, 2004.

Diskussionsforum:

- http://www.die-informatiker.net/