Objektorientierte Programmierung

Einführung in Java und Aufbau von Java-Programmen

Prof. Dr. Ulrike Hammerschall
Fakultät für Informatik und Mathematik

Scope der Lehrveranstaltung

- Voraussetzungen:
  - Sie haben eine Grundidee was Programmierung ist und was man unter einem Softwareprogramm versteht.
  - Sie haben bereits erste Programmiererfahrung mit Python.
  - Sie haben bereits erste Erfahrung mit einer Softwareentwicklungsumgebung (z.B. PyCharm)
- Neue Themen:
  - Einführung in die typisierte Sprache Java
  - Einführung in das Paradigma der Objektorientierung
  - Modellierung objektorientierter Programme mit UML Klassendiagrammen
  - Entwicklung objektorientierter Programme mit Java
  - Umgang mit einer typischen Entwicklungsinfrastruktur
  - Projektarbeit in kleinen Teams
  - Unit-Tests von Java-Klassen mit JUnit
  - Statische Code Analyse mit Checkstyle

Softwareentwicklung aus der Vogelperspektive

Vom Problem zur Lösung:
Entwicklung eines möglichen
Softwaresystems zur Lösung des Problems.

Das **Problem**: wir brauchen ein Softwaresystem, welches ….

Die **Lösung**: ein Softwaresystem geschrieben in einer konkreten Programmiersprache, welches…

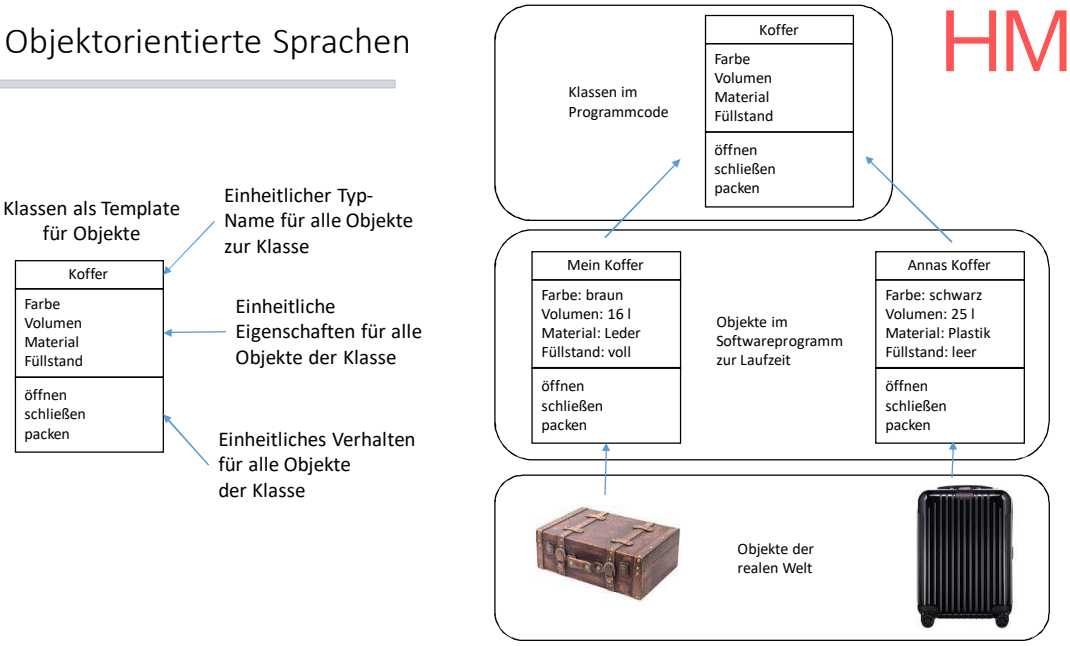
Klassen von Programmiersprachen

- Es gibt verschiedene Klassen von Programmiersprachen
  - Imperative/prozedurale Programmiersprachen (z.B. Fortran, Cobol, C, Pascal)
  - Objektorientierte Programmiersprachen (z.B. Java, C++, C#)
  - Funktionale Programmiersprachen (z.B. Lisp, Haskell, F#, Scala, XSLT)
  - Script-Sprachen (z.B. Javascript, PHP, Python, Perl)
  - *Deskriptive Sprachen (z.B. CSS, HTML, XML, JSON)*
- Der Trend geht heute zu hybriden Sprachen: d.h. Sprachen, die Konzepte verschiedener Klassen unterstützen.

Objektorientierte Sprachen

- Objektorientierte Sprachen bilden die Konzepte der realen Welt auf Objekte und Beziehungen zwischen Objekten ab.
- Die Konzepte können real existieren (z.B. ein Koffer, eine Person) oder virtuell (z.B. ein Bankkonto, ein Kredit) sein.
- Objekte sind eindeutig durch ihre Werte identifizierbar (z.B. ein Bankkonto hat eine Kontonummer, eine Person hat einen Namen).
- Ähnliche Objekte mit ähnlichen Beziehungen werden zu Klassen zusammengefasst.
- Die Klasse definiert die Struktur ihrer Objekte, d.h. welche Eigenschaften und welches Verhalten die Objekte haben.
- Die Klasse dient als Vorlage für alle zugehörigen Objekte. Sie kennt noch keine Werte.
- Die Algorithmen zur Programmdurchführung werden durch das Zusammenspiel von Objekten nachgebildet.

Objektorientierte Sprachen



Die Programmiersprache Java - Historie

- 1995 Erste Urversion von Sun Microsystems veröffentlicht.
  - 1998: Java 1.2: die erste ernstzunehmende Version (J2SE)
  - 2004: Java 1.5 -> Java 5: umfassende Überarbeitung und Erweiterung des Sprachumfangs
- 2010: Die Firma Oracle kauft Sun Microsystems und entwickelt Java als quelloffene Software weiter.
  - 2014: Java 8: Einführung funktionaler Aspekte in Java, sowie viele weitere Überarbeitungen und Erweiterungen.
  - 2018: Java 11: Zum ersten Mal kostenpflichtig für den produktiven Einsatz. In Entwicklungs- und Testumgebungen weiterhin frei, allerdings sehr häufige Versionswechsel.
- OpenJDK:
  - Offizielle frei Implementierung der Java Plattform (Standard Edition).
  - Entstand als Abspaltung von Java 6
  - Aktuelle Version: OpenJDK 19
  - Version für das Praktikum: OpenJDK 18
- Aufgrund der lizenzrechtlichen Änderungen verwendet die Fakultät aktuell OpenJDK.

Java und Python im Vergleich

Java

- **Objektorientierte** Sprache (mit funktionalen Elementen)
- **Statische Typisierung**: Jede Variable erhält bei Definition einen eindeutigen unveränderlichen Datentyp. Die Typprüfung erfolgt durch den Compiler.
- Der **Compiler** übersetzt den Source-Code in Maschinenlesbaren Bytecode.
- Die Ausführung des Bytecodes erfolgt durch den **Interpreter** in der **Java Runtime (JVM)**.

Python

- **Skriptsprache**, die u.a. auch funktionale Programmierung und Objektorientierung unterstützt
- **Dynamische Typisierung**: die Typprüfung erfolgt zur Laufzeit durch den Interpreter.
- Es gibt keinen expliziten Compiler. Der **Interpreter** führt direkt den Source-Code aus.

Grundstruktur einer Java-Klasse

Klasse: Jede Klasse repräsentiert ein Konzept des zu lösenden Problems mit seinem Verhalten und seinen Zuständen.

Objektvariablen: Repräsentieren spezifische Eigenschaften einer Klasse. Eine Klasse kann beliebig viele, auch keine Objektvariablen enthalten.

Methoden: beschreiben das Verhalten einer Klasse. Typischerweise machen Methoden Berechnungen und / oder verändern Werte von Objektvariablen. Eine Klasse kann beliebig viele, auch keine Methoden enthalten.

Beispiel 1: Die Klasse Konto (Account)

Klasse

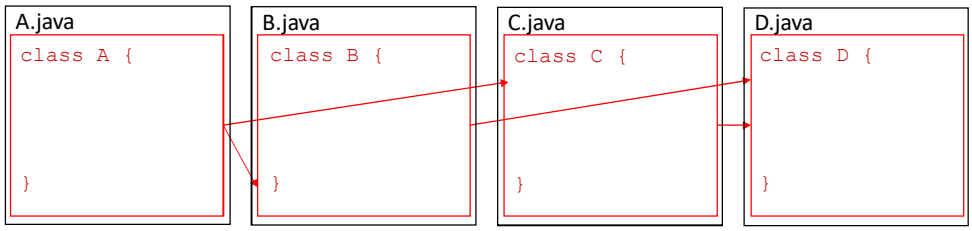
```
class Account {

    // aktueller Kontostand
    private double balance = 0;

    // Einzahlungen in das Konto
    void deposit(double amount) {
        balance = balance + amount;
    }

    // Abfrage des aktuellen Kontostands
    double getBalance() {
        return balance;
    }

}
```

Objektvariablen

Methoden

Grundstruktur eines Java-Programms - 1

- Ein Programm besteht aus beliebig vielen (jedoch mindestens einer) Klassen mit Beziehungen untereinander. Jede Klasse im Programm hat ihren individuellen Namen, ihre Objektvariablen und ihre Methoden.
- Jede Klasse wird in einer eigenen Datei definiert. Der Dateiname entspricht dem Klassennamen mit der Endung .java

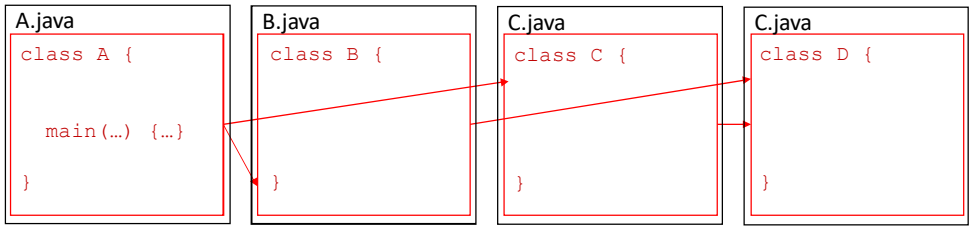


Grundstruktur eines Java-Programms - 2

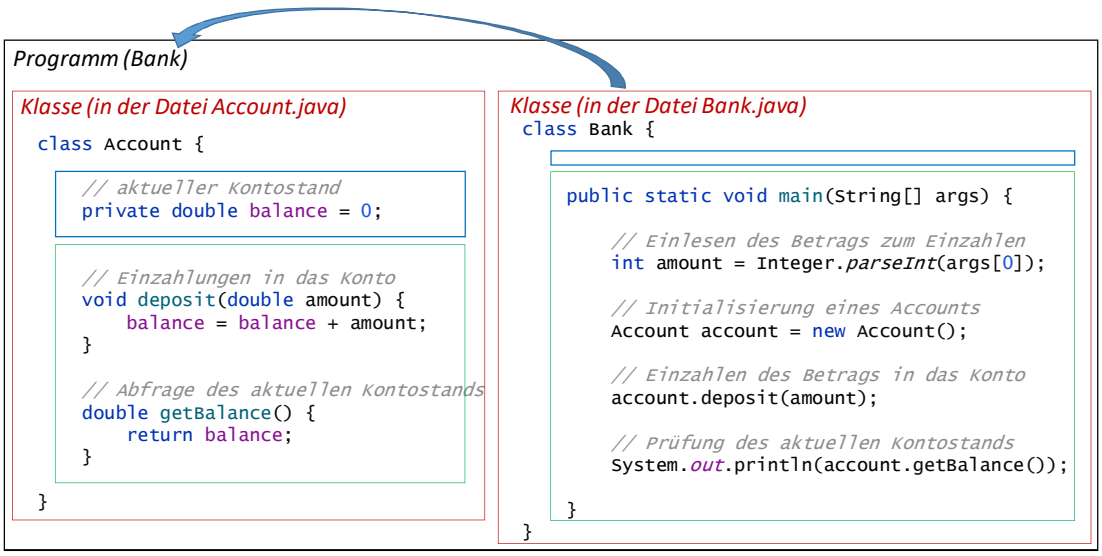
Genau eine Klasse im Programm enthält eine *main*-Methode:
- Klassenname der Klasse mit *main*-Methode entspricht dem Programmnamen.
- Über diese Klasse wird das Programm gestartet.

Die Signatur der *main*-Methode ist fest vorgegeben:

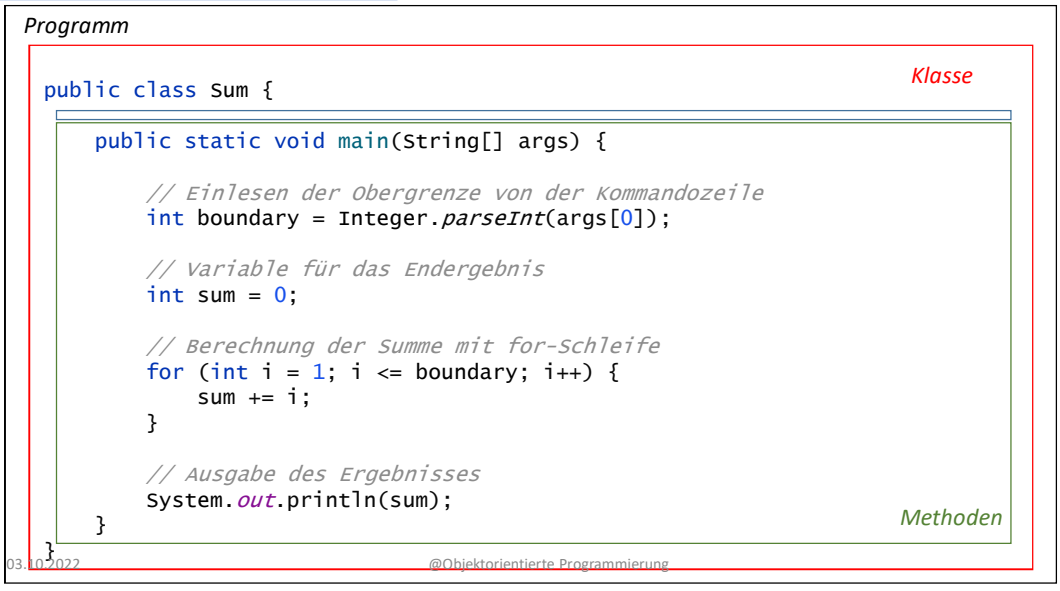
```
public static void main(String[] args) {…}
```



Beispiel 2: Eine kleine Bankanwendung



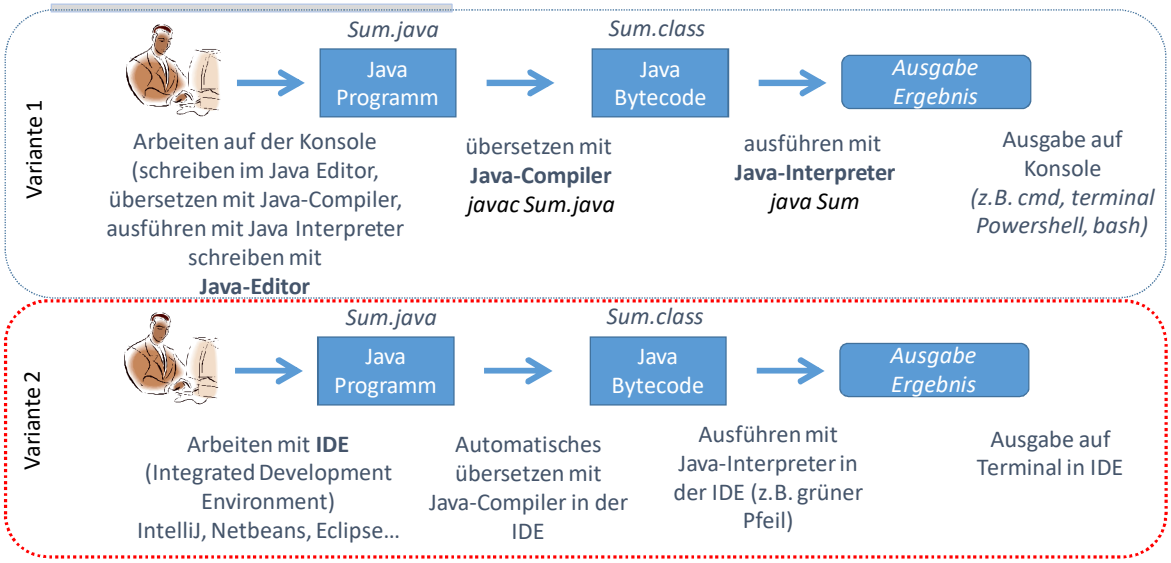
Beispiel 3: Einfaches Programm zur Summenberechnung



Java Entwicklungsumgebung und Werkzeuge

- Java Development Kit (JDK)
  - Liefert Werkzeuge für die Entwicklung von Java Programmen (z.B. javac, java, javadoc…)
- Java Runtime Environment (JRE)
  - Teil des JDK oder eigenständig verfügbar
  - Schnittstelle zum realen Betriebssystem. Simuliert für Programme die Dienste des Betriebssystems.
  - Ablaufumgebung für den Interpreter zur Abarbeitung der Anweisungen im Bytecode.
  - Speicherverwaltung (Garbage Collector).
  - Bereitstellung von Bibliotheken.
- Achtung:
  - JDK und JRE können prinzipiell getrennt heruntergeladen werden. Sie benötigen für die Programmierung aber immer beides -> Immer JDK herunterladen!

Übersetzen und Ausführen von Java-Programmen



Zusammenfassung

- Ziel der Softwareentwicklung ist die Erstellung von Softwaresystemen zur Lösung spezifischer Problemstellungen in der Praxis.
- Als Programmieren bezeichnet man den Prozess zur Beschreibung eines Lösungsalgorithmus mit Hilfe einer Programmiersprachen.
- Es gibt unterschiedliche Klassen von Programmiersprachen mit verschiedenen Problemlösungsstrategien und Konzepten.
- Java ist eine klassische objektorientierte Sprache. Sie unterstützt statische Typisierung.
- Ein Java-Programm besteht immer aus einer oder mehreren Klassen. Die Klassen werden mit einem Compiler übersetzt und mit einem Interpreter ausgeführt.
- Zur Laufzeit werden zu den Klassen Objekte erzeugt. Diese interagieren miteinander über Methodenaufrufe.