PROGRAMMIEREN I

MATTHIAS BERG-NEELS

VORSTELLUNGSRUNDE

- Name
- Firma
- Guess what?

ANMERKUNGEN ZUR VORLESUNG

- Skript
  - PDF Download
- Repositories: https://gitlab.mubn.de

KAPITELÜBERSICHT - PROGRAMMIEREN 1

1. Einführung
2. Grundlagen von Java
3. Datentypen
4. Ausdrücke und Anweisungen
5. Okjektorientierung
6. Vererbung
7. Interfaces

QUELLEN & LITERATURVERZEICHNIS

- **basierend auf dem Skript "Programmieren 1 + 2" von Michael Lang**

- BALZERT, HEIDE: Lehrbuch der Objektmodellierung - Analyse und Entwurf. Spektrum Akademischer Verlag, 2. Auflage, 2005. ISBN 3-8274-1162-9
- HÄUSLEIN, ANDREAS: Systemanalyse - Grundlagen, Techniken, Notierungen. VDE Verlag, 2004. ISBN 3-8007-2715-3
- HITZ, M., KAPPEL, G., KAPSAMMER, E. und RETSCHITZEGGER, W.: UML@Work - Objektorientierte Modellierung mit UML 2. dpunkt.verlag, 3., aktualisierte und überarbeitete Auflage, 2005. ISBN 3-89864-261-5
- HOLEY, T., WELTER, G. und WIEDEMANN, A.: Wirtschaftsinformatik. Kiehl Verlag, 2004. ISBN 3-470-52791-1
- RUPP, CHRIS / SOPHIST GROUP: Systemanalyse kompakt. Elsevier Spektrum Akademischer Verlag, 2004. ISBN 3-8274-1509-8
- STAHLKNECHT, P. und HASENKAMP, U.: Arbeitsbuch Wirtschaftsinformatik. Springer Verlag, 4. Auflage, 2006. ISBN 3-540-26361-6
- STAHLKNECHT, P. und HASENKAMP, U.: Einführung in die Wirtschaftsinformatik. Springer Verlag, 10. Auflage, 2002. ISBN 3-540-41986-1
- ZUSER, W., BIFFL, S., GRECHENING, T. und KÖHLE, M.: Software Engineering mit UML und dem Unified Process. Pearson Studium, 2001. ISBN 3-8273-7027-2

KAPITEL 1
EINFÜHRUNG

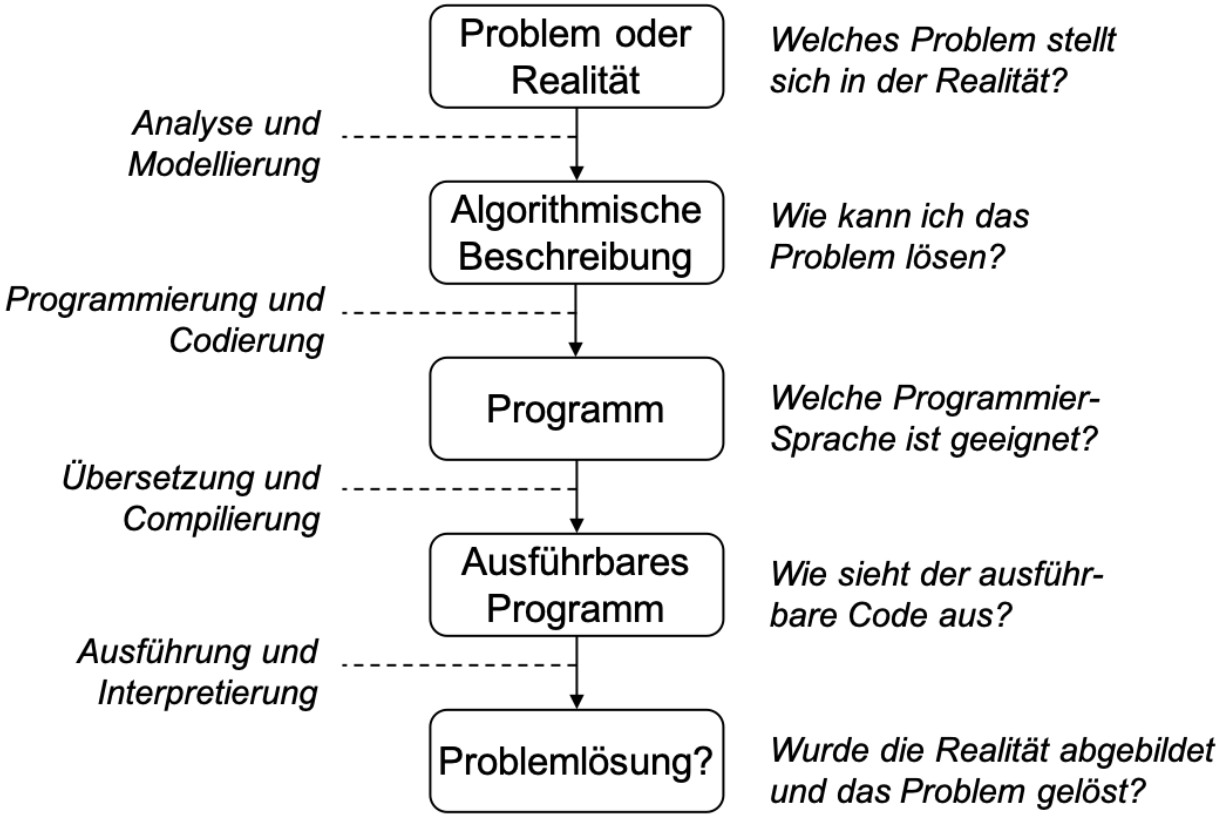
ÜBERSICHT

LERNZIELE

- Sie können den Begriff Algorithmus definieren
- Sie kennen die Eigenschaften und Bestandteile von Algorithmen
- Sie können die Grundbegriffe der Programmierung nennen und einsetzen
- Sie kennen unterschiedliche Darstellungsformen von Algorithmen
- Sie können einfache Algorithmen in Form von Pseudocode, Programmablaufplänen und Struktogrammen darstellen

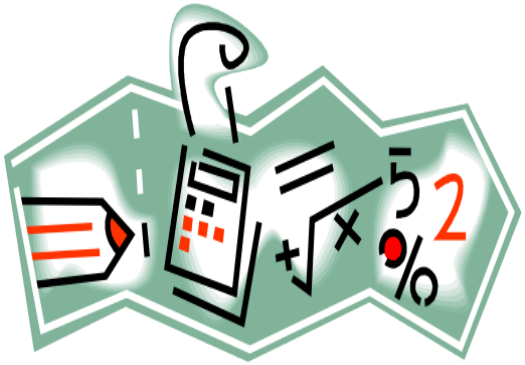
ZIEL DER PROGRAMMIERUNG

Umsetzung eines gegebenen oder selbstentwickelten Algorithmus in ein lauffähiges Computerprogramm



BEISPIELE FÜR ALGORITHMEN

- Bedienungsanleitungen
- Bauanleitungen
- Kochrezepte
- mathematische Problemstellungen
- Such- und Sortieralgorithmen



BEISPIEL: EIN KOCHREZEPT

Zutaten

- 500 g Hackfleisch vom Rind
- 2 große Zwiebeln, fein gehackt
- 4 Stangen Staudensellerie, fein gehackt
- 150 g Speck, fein gehackt
- 2 Zehen Knoblauch, fein gehackt
- 400 g geschälte Tomaten
- 300 ml Fond (Bratenfond)
- 1 Paprikaschote, rot, klein gewürfelt
- 1 Gewürznelke
- 1 Lorbeerblatt
- ½ TL Oregano
- ½ TL Muskat (gemahlen)
- 3 EL Öl (Oliven)
- 40 g Butter
- 150 ml Wein, rot, trocken
- 1 Chilischote, zerkleinert, getrocknet

Zubereitung

Butter und Öl in einer Pfanne (besser Bräter) erhitzen. Zwiebeln, Sellerie, Speck und Paprika gut anbraten. Die Sachen aus der Pfanne nehmen und zur Seite stellen. Den Knoblauch leicht anrösten und dann das Hackfleisch dazugeben, alles gut anbraten. Die Sachen wieder dazugeben und alles ca.10 Min. köcheln lassen. Jetzt alle übrigen Zutaten und Gewürze dazugeben und das Ganze ca. 45 Min. (bei geschlossenem Deckel) köcheln lassen (gelegentlich abschmecken und ggf. nachwürzen). Über die fertig gegarten Spaghetti geben.

PROBLEME BEI DIESEM BEISPIEL

- Zutaten sind bereits vorbereitet
- Spaghetti fehlen bei Zutaten
- Beschreibung für Zubereitung der Spaghetti fehlt
- ungenaue Aussagen
  - fein gehackt
  - klein gewürfelt
  - erhitzen
  - gut anbraten
  - leicht anrösten
  - …
- Pfanne (besser Bräter)

KURZ: Die Beschreibung lässt Raum für individuelle Entscheidungen und Interpretationen.

VERBESSERUNGSMÖGLICHKEITEN DES REZEPTS

- komplette Eliminierung von individuellen Interpretations-spielräumen
- vollständige Beschreibung der Arbeitsschritte inkl. Vor-bereitung und der Kochanweisung für die Spaghetti
- vollständige Angabe der Zutaten präzise Angaben bei den Aussagen
  - fein gehackt ⇒ gewürfelt, 2 mm ≤ Kantenlänge ≤ 3 mm(Zwiebeln und Speck)
  - fein gehackt ⇒ gewürfelt, 0,8 mm ≤ Kantenlänge ≤ 1 mm(Knoblauch)
  - klein gewürfelt ⇒ 4 mm ≤ Kantenlänge ≤ 6 mm
  - erhitzen ⇒ 120 °C < Temperatur < 125 °C
  - gut anbraten ⇒ Farbe der Zwiebeln entspricht dem RGB-Wert CC3300
  - leicht anrösten ⇒ Farbe des Knoblauchs entspricht dem RGB-Wert CC6600

ALGORITHMUS: DEFINITION

DUDEN

> *Al|go|rith|mus, der; , ...men [mlat. algorismus = Art der indischen Rechenkunst, in Anlehnung an griech. arithmós = Zahl entstellt aus dem Namen des pers.- arab. Mathematikers Al-Hwarizmi, gest. nach 846] (Math., Datenverarb.): Verfahren zur schrittweisen Umformung von Zeichenreihen; Rechenvorgang nach einem bestimmten [sich wiederholenden] Schema.*

INFORMATIK

> *Ein Algorithmus ist eine präzise (d.h. in einer festgelegten Sprache abgefasste) endliche Beschreibung eines allgemeinen Verfahrens unter Verwendung ausführbarer elementarer (Verarbeitungs-)Schritte.*

ALGORITHMUS: EIGENSCHAFTEN

TERMINIERUNG

- bricht nach endlich vielen Schritten ab

DETERMINISMUS

- legt die „Wahlfreiheit“ fest
- deterministischer Ablauf
  - legt eindeutige Vorgabe der Schrittfolge der auszuführenden Schritte fest
- determiniertes Ergebnis
  - wird immer dann geliefert, wenn bei vorgegebener Eingabe ein eindeutiges Ergebnis geliefert wird - auch bei mehrfacher Durchführung mit denselben Eingabeparametern

---

ALGORITHMUS: BESTANDTEILE

- elementare Operationen (Ausdrücke und Anweisungen)
  Berechne 5 plus 7
- sequenzielle Ausführung
  Berechne 10 minus 3, dann multipliziere das Ergebnis mit 4
- parallele Ausführung
  Du rechnest Aufgabe 1 und ich rechne Aufgabe 2
- bedingte Ausführung
  Wenn Du Aufgabe 1 gelöst hast, dann beginne mit Aufgabe 2
- Schleife
  Rechne Aufgabe 1, bis Du das richtige Ergebnis bekommst
- Unterprogramm
  Rechne Aufgabe 1 anhand der Lösung auf Seite 106
- Variablen und Konstanten

GRUNDBEGRIFFE DER PROGRAMMIERUNG

Ausdruck

- Kombination von Operanden und Operatoren als "Vorschrift" zur Berechnung eines Werts
- liefert immer einen Wert (Ergebniswert) ab
- Beispiel:1 / x

Anweisung

- Kombination von Ausdrücken und Methoden als "Vorschrift" zurAusführung einer Aktion
- Beispiele:
  x = 5 Wertzuweisung
  y = 1 / x Wertzuweisung
  print(x) Ausgabeanweisung (Methodenaufruf "Drucke x")

GRUNDBEGRIFFE DER PROGRAMMIERUNG

Sequenz

- bildet eine zeitliche Abfolge von Anweisungen
- einzelne Schritte werden durchnummeriert oder es wird zum Abschluss der Sequenz ein Semikolon gesetzt

Bedingte Anweisung

- es werden Bedingungen auf Ihre Richtigkeit geprüft
- für wahre und falsche Aussagen in der Bedingung können unterschiedliche Anweisungen ausgeführt werden

Schleifen

- bestimmte Anweisungen werden wiederholt, bis eine definierte Endbedingung erfüllt wird
- Unterscheidung in drei Schleifenarten

GRUNDBEGRIFFE DER PROGRAMMIERUNG

Unterprogramme

- beinhaltet einen Teilalgorithmus
- dieser Teilalgorithmus kann in mehreren Algorithmen wieder verwendet werden

Variablen

- „Platzhalter“ für einen konkreten Wert
- sind von einem bestimmten Datentyp können ihren Wert ändern

Konstanten

- haben einen festen Wert
- sind von einem bestimmten Datentyp
- können ihren Wert NICHT ändern

DARSTELLUNGSFORMEN VON ALGORITHMEN

Pseudocode

- nahe an den Konstrukten verbreiteter Programmiersprachen
- Verwendung spezieller englischer Begriffe aus dem Alltag
- Begriffe haben eine festgelegte Bedeutung

Programmablaufpläne

- genormt nach DIN 66001
- Ursprung in der linearen Programmierung
- nur für kleinere Programme geeignet (Übersichtlichkeit)

Nassi-Schneiderman-Diagramme (Struktogramme)

- Entstehung 1973
- Darstellung genormt nach DIN 66261 im Jahr 1985
- überwiegender Einsatz in der prozeduralen Programmierung

PSEUDOCODE

Sequenz

- Alternative 1: Schritte werden durchnummeriert: 1, 2, 3, …
- Alternative 2: Abschluss der Sequenz durch Semikolon
- Vorteil Alternative 1: Verfeinerung einzelner Schritte: 2.1, 2.2, …

Bedingte Anweisung

- es werden Bedingungen auf Ihre Richtigkeit geprüft
- Alternative 1: falls Bedingung dann Schritt
- Alternative 2: falls Bedingung dann Schritt A sonst Schritt B

Schleifen

- kopfgesteuert: solange Bedingung wahr führe aus Schritte
- fußgesteuert: wiederhole Schritte bis Bedingung wahr
- Zählschleife: wiederhole für Zahlenbereich Arbeitsschritte

BEISPIEL: PSEUDOCODE

SEQUENZ

1. Koche Wasser
2. Gib Kaffeepulver in Tasse
3. Fülle Wasser in Tasse

BEDINGTE ANWEISUNG

falls Ampel rot oder gelb
  dann stoppe
  sonst fahre weiter

falls Ampel ausgefallen
  dann fahre vorsichtig weiter
  sonst falls Ampel grün
    dann fahre weiter
  sonst stopp

SCHLEIFEN

solange Liste nicht erschöpft
  führe aus
    Gib nächste Zahl aus der Liste aus

wiederhole
  Gib nächste Zahl aus derListe aus
bis Liste erschöpft

wiederhole für 5 bis 10
  Gib nächste Zahl aus derListe aus

STRUKTOGRAMME

Sequenz

- Eingabe: 2 Zahlen ohne Kommastellen
- Addiere die beiden Zahlen
- Ausgabe: Ergebnis der Addition

Bedingte Anweisung

- Eingabe: 2 Zahlen ohne Kommastellen
- Zahl 1 > Zahl 2

Ausgabe: Zahl 1 ist größer als Zahl 2	Ausgabe: Zahl 2 ist größer als Zahl 1

Mehrfachverzweigung

- Eingabe: 2 Zahlen und 1 Operator
- Operator ist

+	-	
Ergebnis = Zahl 1 + Zahl 2	Ergebnis = Zahl 1 - Zahl 2	Operation nicht möglich

- Ausgabe: 2 Zahlen, Operator, Ergebnis

Kopfgesteuerte Schleife

- Zähler auf 0 setzen
- Solange Zähler kleiner 100 ist
  - Zähler um 1 erhöhen

Fußgesteuerte Schleife

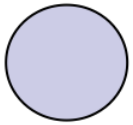
- Zähler auf 0 setzen
  - Zähler um 1 erhöhen
- bis Zähler gleich 100 ist

Zählschleife

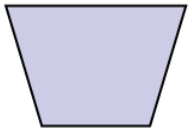
- Für Zähler von 1 bis 100
  - Ausgabe: Zähler
- Ausgabe: „Unser Rechner kann Zählen“

PROGRAMMABLAUFPLAN

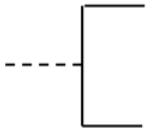
Allgemeine Verarbeitung (einschl. Ein- und Ausgabe)



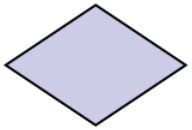
Verbindungsstelle



manuelle Verarbeitung (einschl. Ein- und Ausgabe)



Bemerkung (erläuternder Text)

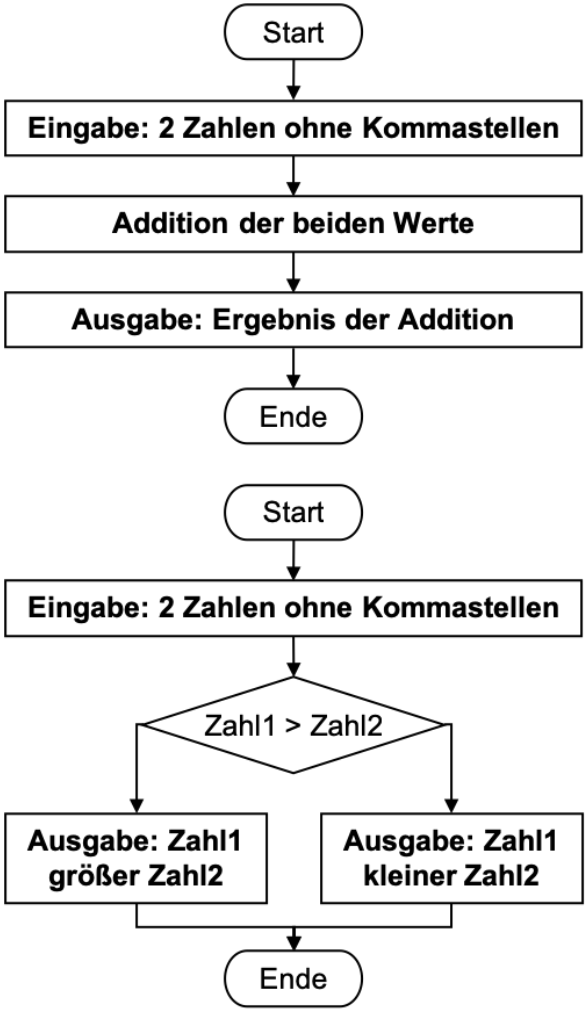


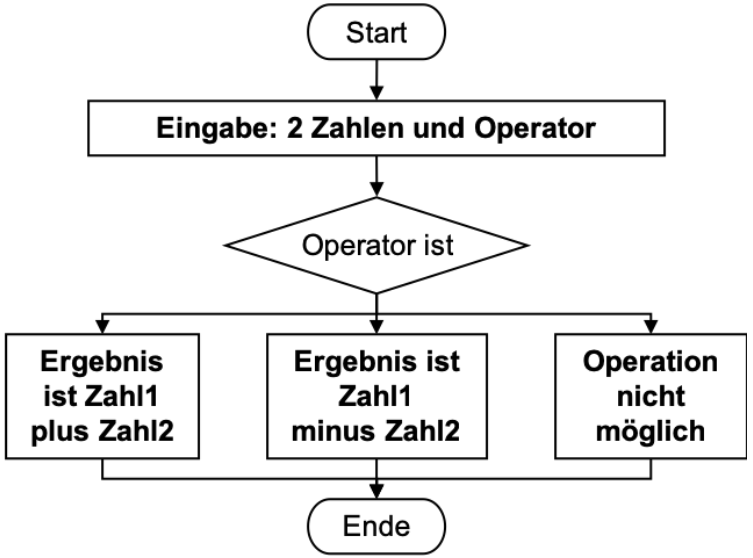
Verzweigung

Verbindung (Verarbeitungsfolge)

Grenzstelle (zur Umwelt)

BEISPIEL: PROGRAMMABLAUFPLAN





BEISPIEL: MATHEMATISCHE PROBLEMSTELLUNG

gesucht sei ggT(969,627) durch Anwendung des Euklidschen Algorithmus

342 = 969 - 627
285 = 627 - 342
57 = 342 - 285
228 = 285 - 57
171 = 228 - 57
114 = 171 - 57
57 = 114 - 57
0 = 57 - 57

allgemeingültige Formulierung mit den Variablen x und y: ggT(x,y)

solange x ungleich y führe aus
falls x größer y dann ergibt sich x aus x minus y
falls y größer x dann ergibt sich y aus y minus x
gib das Ergebnis x aus

Übung: Erstellen Sie zum beschriebenen des euklidschen Algorithmus ein Struktogramm und einen Programmablaufplan.

KAPITEL 2
GRUNDLAGEN VON JAVA

ÜBERSICHT

1. Einführung
2. **Grundlagen von Java**
3. Datentypen
4. Ausdrücke und Anweisungen
5. Okjektorientierung
6. Vererbung
7. Interfaces

LERNZIELE

- Sie können die Eigenschaften der Programmiersprache Java beschreiben
- Sie kennen die Aufgaben von Compiler, Linker und Interpreter
- Sie können das Zusammenspiel von Bytecode und der Java Virtual Machine erläutern
- Sie können die wesentlichen Java-Tools benennen und ihre Aufgabe beschreiben
- Sie können das Paketkonzept in Java erläutern
- Sie kennen die wesentlichen Systemvariablen im Umfeld von Java
- Sie können Eclipse als Java-Entwicklungsumgebung einsetzen

VORBEREITUNG

- [prüfen] Installieren Sie das Java Developer Kit, falls notwendig. JDK Download von Oracle
- Aktivieren Sie eine Studierenden-Lizens und installieren Sie IntelliJ Jetbrains Free License for faculty members - DHBW-Email benötigt

KLASSIFIZIERUNG VON SOFTWARE

- Programme zur Steuerung der Verarbeitungsprozesse, der Übertragungsprozesse und der Speicherungsprozesse in Computern
- unterschiedliche Softwarearten
  - Systemsoftware
    - grundlegende Dienste für andere Programme
    - Steuerung des Computersystems (Hardware)
    - Ablaufsteuerung anderer Programme
  - Entwicklungssoftware
    - Erstellung und Modifikation von Programmen
    - Übersetzungsprogramme für Programmiersprachen
  - Anwendungssoftware
    - Programme zur Verarbeitung der Daten
    - Unterhaltungssoftware
    - Spiele

EIGENSCHAFTEN VON JAVA

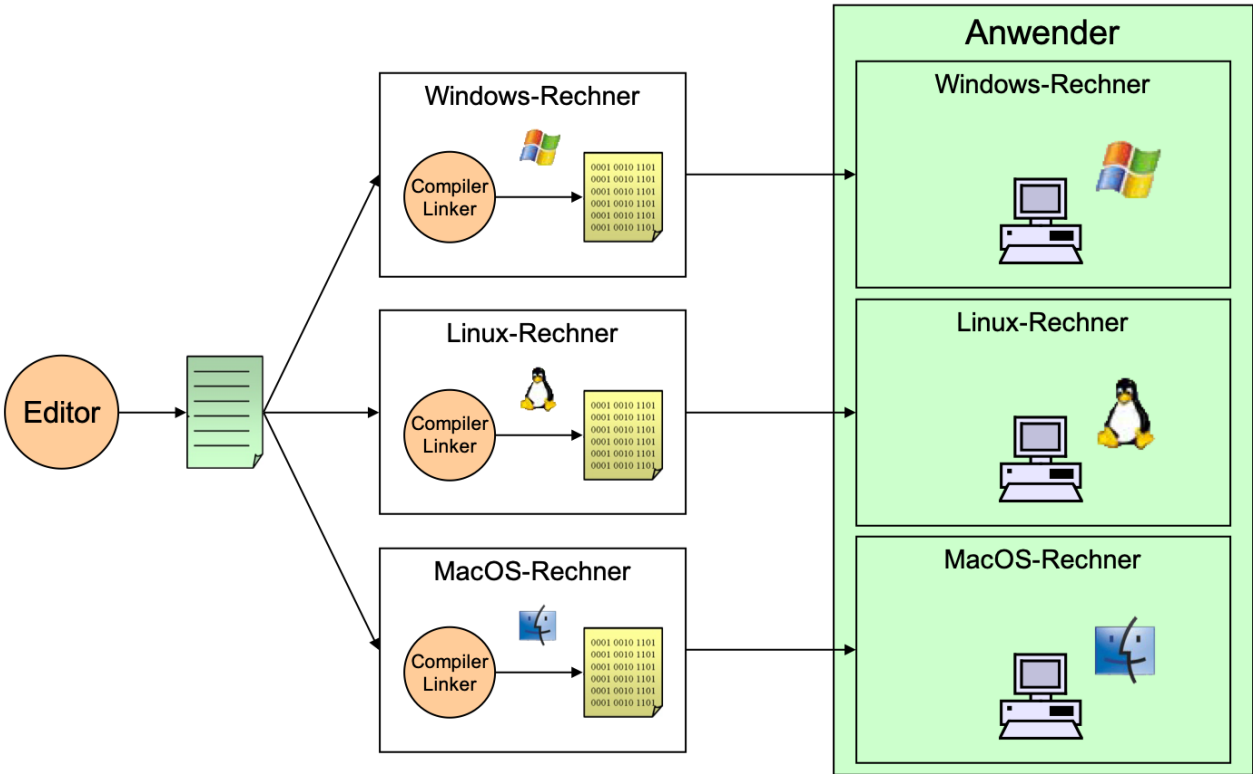
- vollständig objektorientiert
  - ohne prozedurale Altlasten
- unkompliziert - einfach und leicht erlernbar
  - Syntax ähnlich zu C, C++
  - Beschränkung auf das Notwendigste
  - keine Pointer
  - keine Header-Dateien
  - keine Präprozessor-Anweisungen
  - keine Mehrfachvererbung
- plattformunabhängig (architekturneutral)
  - übersetzte Java-Programme (Bytecode) sind auf jeder javafähigen Plattform ausführbar
  - fest definierter Wertebereich für Zahlen

EIGENSCHAFTEN VON JAVA

- sicher
  - keine direkten Speicherzugriffe mit * Pointerarithmetik
  - strenge Typüberprüfung
  - keine Programmierung mit Sprachverletzung
- robust
  - keine Rechnerabstürze durch Programmierfehler
  - Überprüfung der Speicherzugriffe
  - Ausnahmeroutinen zur Fehlerbehandlung
- multithreaded
  - parallele Ausführung von Programmteilen
- internetfähig
  - Java-Applets sind über das Internet verteilbar und können lokal auf dem Rechner ausgeführt werden

UMWANDLUNG VON QUELLCODE IN MASCHINENSPRACHE

COMPILER


Editor
Windows-Rechner
Compiler
Linker
0001 0010 1101
Linux-Rechner
Compiler
Linker
0001 0010 1101
MacOS-Rechner
Compiler
Linker
0001 0010 1101
Anwender
Windows-Rechner
Linux-Rechner
MacOS-Rechner

FUNKTIONSWEISE EINES COMPILER

- übersetzt ein Computerprogramm aus einer Quellsprache in ein semantisch äquivalentes Programm einer Zielsprache
- Aufbau eines Compilers in 2 Phasen
  - Analysephase
    - lexikalische Analyse zerteilt Code in zusammengehörende Token
    - syntaktische Analyse überprüft auf formale Richtigkeit
    - semantische Analyse überprüft die logischen Rahmenbedingungen
  - Synthesephase
    - Zwischencodeerzeugung liefert die Basis für die Optimierung
    - Programmoptimierung auf Basis des maschinennahen Zwischencodes
    - Codegenerierung erzeugt den Programmcode der Zielsprache

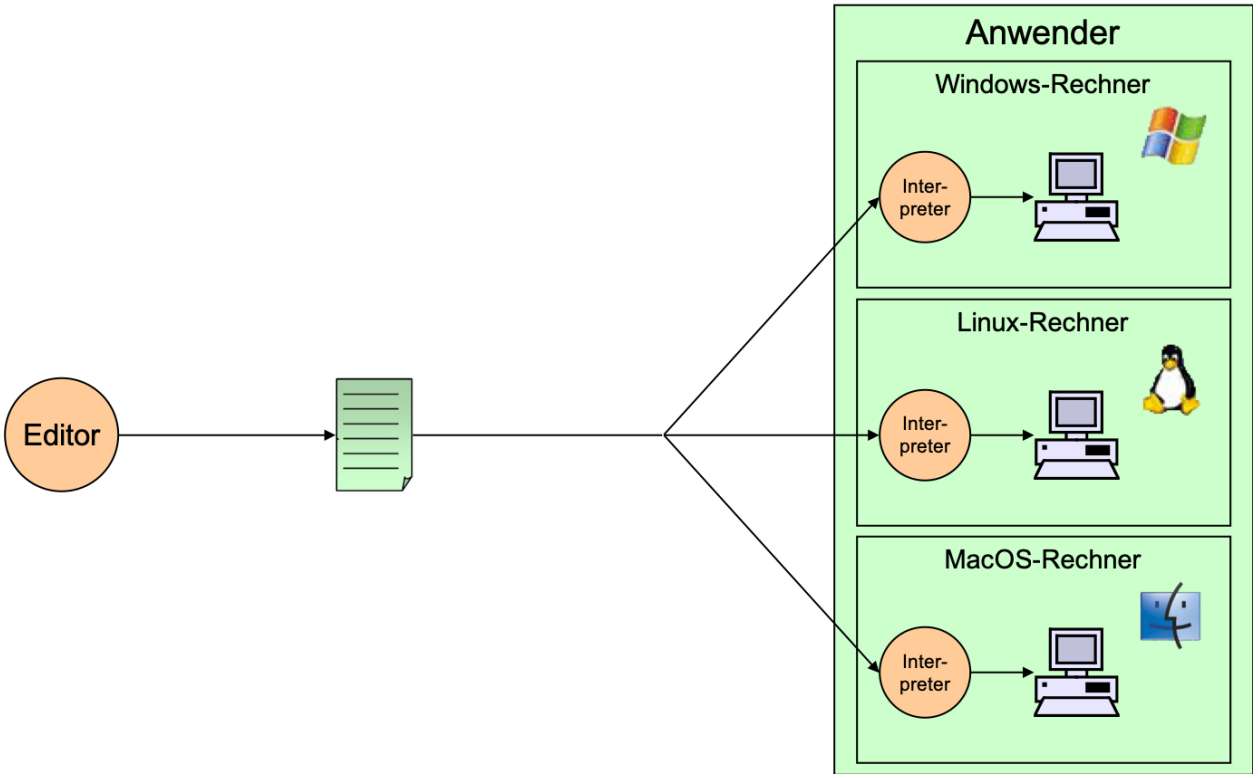
ARTEN VON COMPILERN

- Native Compiler erzeugt Code für die Plattform, auf der er läuft
- Cross-Compiler erzeugt Code für andere Plattformen
- One-pass-Compiler erzeugt den Code in einem Durchlauf
- Multi-pass-Compiler erzeugt den Code in mehreren Durchläufen

LINKER

- stellt einzelne Programmmodule zu einem ausführbaren Programm zusammen
- fügt benötigten Code aus Funktionsbibliotheken zum Code des Hauptprogramms hinzu
- statisches Linken
  - beim kompilieren werden benötigte Codings aus Funktionsbibliotheken dem Coding des Hauptprogramms hinzugefügt
- dynamisches Linken
  - zur Laufzeit werden die Codings aus Funktionsbibliotheken o.ä. zum Hauptprogramm hinzugelinkt
    - DLL-Konzept von Windows
    - auch bei Java findet dynamisches Linken statt

INTERPRETER

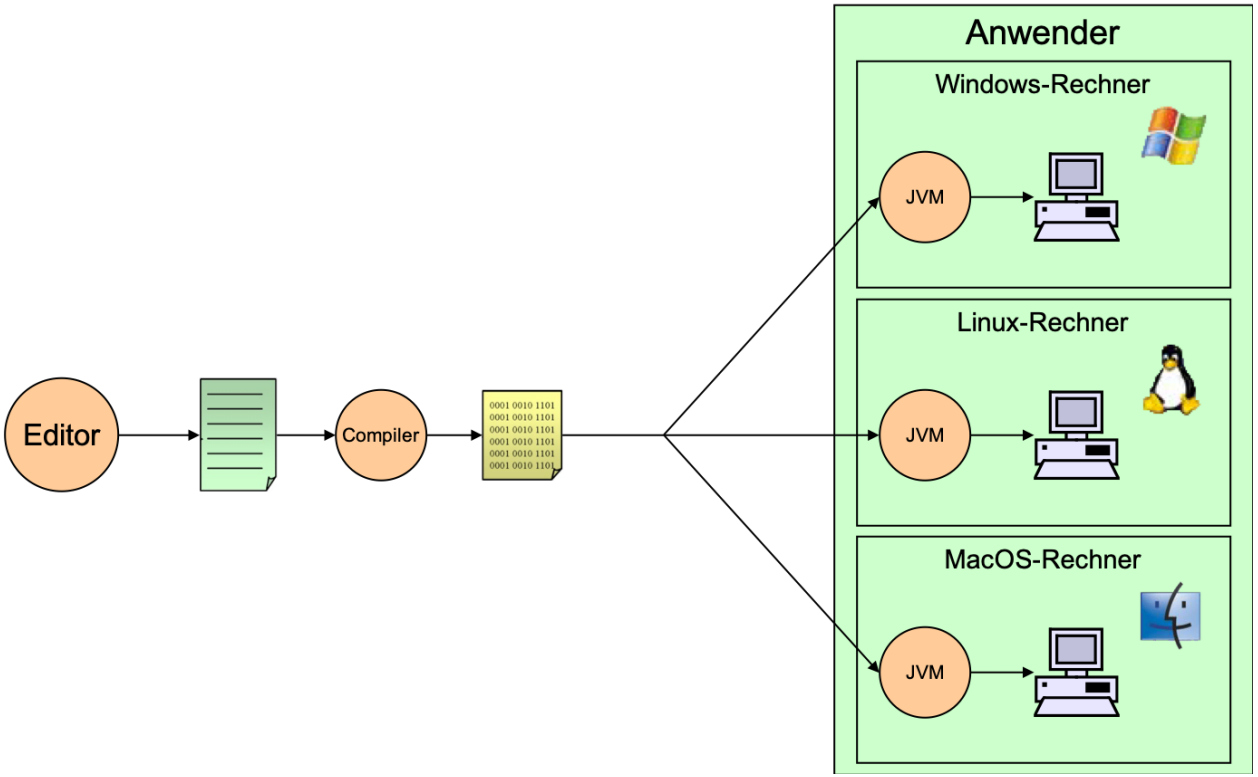


FUNKTIONSWEISE EINES INTERPRETERS

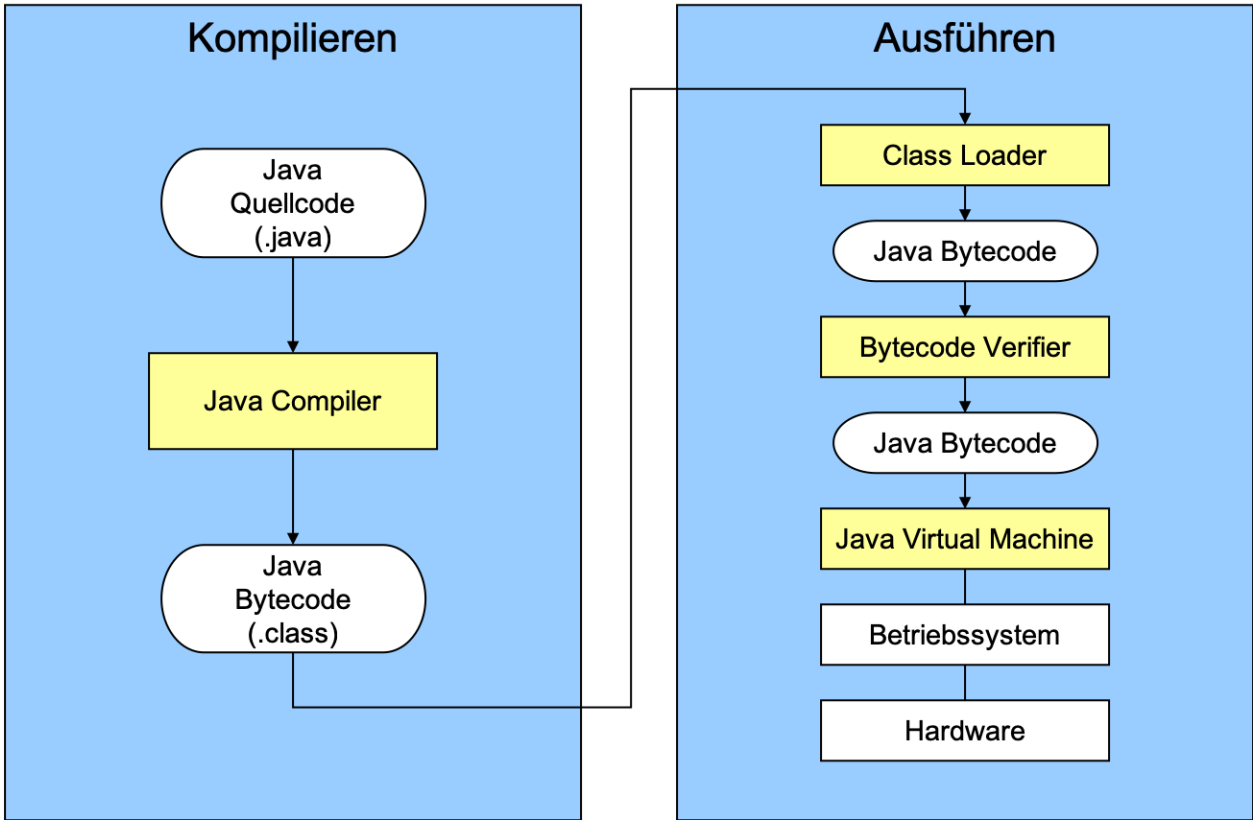
- übersetzt den Quellcode nicht in eine direkt ausführbare Datei
- liest den Quellcode ein, analysiert diesen und führt ihn dann aus
- die Analyse erfolgt also zur Laufzeit des Programms
- auf jeder Rechnerarchitektur lauffähig
- deutlich langsamer als kompilierte Programme
- Möglichkeiten zur Steigerung der Geschwindigkeit
  - Just-In-Time-Compiler
    - zur Laufzeit wird der Quellcode in einen Maschinencode übersetzt
    - Maschinencode wird direkt vom Prozessor ausgeführt
    - mehrfach durchlaufene Programmteile müssen nur ein Mal übersetzt werden
    - nur auf einer bestimmten Rechnerarchitektur lauffähig
  - Bytecode-Interpreter
    - Quellcode wird zur Laufzeit in sog. Bytecode übersetzt
    - Bytecode wird von einem Interpreter (Virtual Machine) ausgeführt
    - Bytecode kann auf verschiedenen Plattformen ausgeführt werden

JAVAQUELLCODE ZU MASCHINENSPRACHE

PLATFORMUNABHÄNGIGKEIT DURCH BYTECODE



BYTECODE UND VIRTUAL MACHINE



WESENTLICHE JAVA-TOOLS

Java-Compiler (javac.exe)

Java virtual Machine (java.exe)

Javadoc

- dient der automatischen Erstellung von Dokumentationen

Class Loader

- lädt Java-Klassen in den Arbeitsspeicher
- bereitet Ausführung von Java-Applikationen vor

Bytecode Verifier

- Prüfung auf syntaktische Korrektheit
- Überprüfung der Klassenhierarchie
- Überprüfung jeder Methode auf strukturelle Gültigkeit
- Datenflussanalyse, um u.a. Typfehler zu vermeiden

WICHTIGE SYSTEMVARIABLEN

PATH

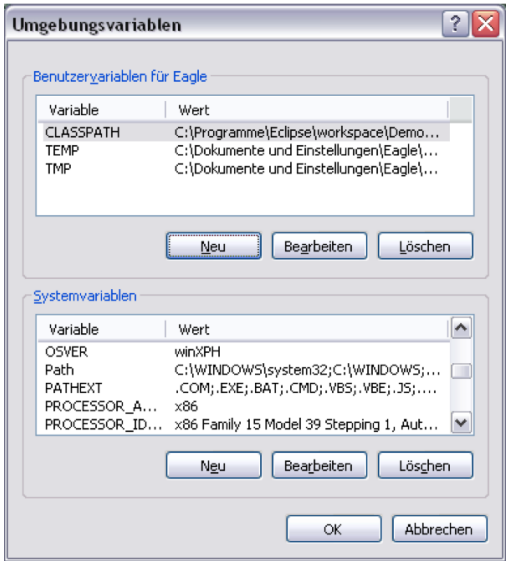
- Verzeichnis, in dem sich dieJAVA-Tools befinden

CLASSPATH

- Verzeichnisse, in denen nachKlassen und Paketen gesucht werden soll

JAVA_HOME

- Verzeichnis, in dem das J2SDKinstalliert wurde



BEISPIEL: HELLO WORLD ÜBER DIE KONSOLE

1. anlegen einer Datei "HelloWorld.java" mit folgendem Quellcode

```
class HelloWorld {

  public static void main (String[] args){
    System.out.println("Hello to the Java World");

  }
}
```

1. Quellcode zu Bytecode kompilieren

```
javac HelloWorld.java
```

1. Ausführen des komplilierten Bytecode in Java Virtial Machine (JVM)

```
java HelloWorld
```

JAVA - KOMMENTARE IM QUELLCODE

Einzeiliger Kommentar

```
// single line comment -
```

Blockkommentar (Mehrzeiliger Kommentar)

```
/*
multi line comment
starting with the signs

ending with the signs
*/
```

Beispiel

```
// HelloWorld class as first small programming example
class HelloWorld {

  // main method as starting point in Code
  public static void main (String[] args){
    /*
    Implementation of main method
    prints out "Hello to the Java World"
    to the console.
    */
    System.out.println("Hello to the Java World");
  }
}
```

EXKURS: JAVADOC

GENERIERUNG VON DOKUMENTATION AUS JAVAQUELLCODE UND KOMMENTAREN

- Generiert Dokumentation anhand des Quellcodes
- Kommentare mit spezieller Formatierung ("/** ") werden berücksichtig
- Resultat: Dokumentation im HTML-Format

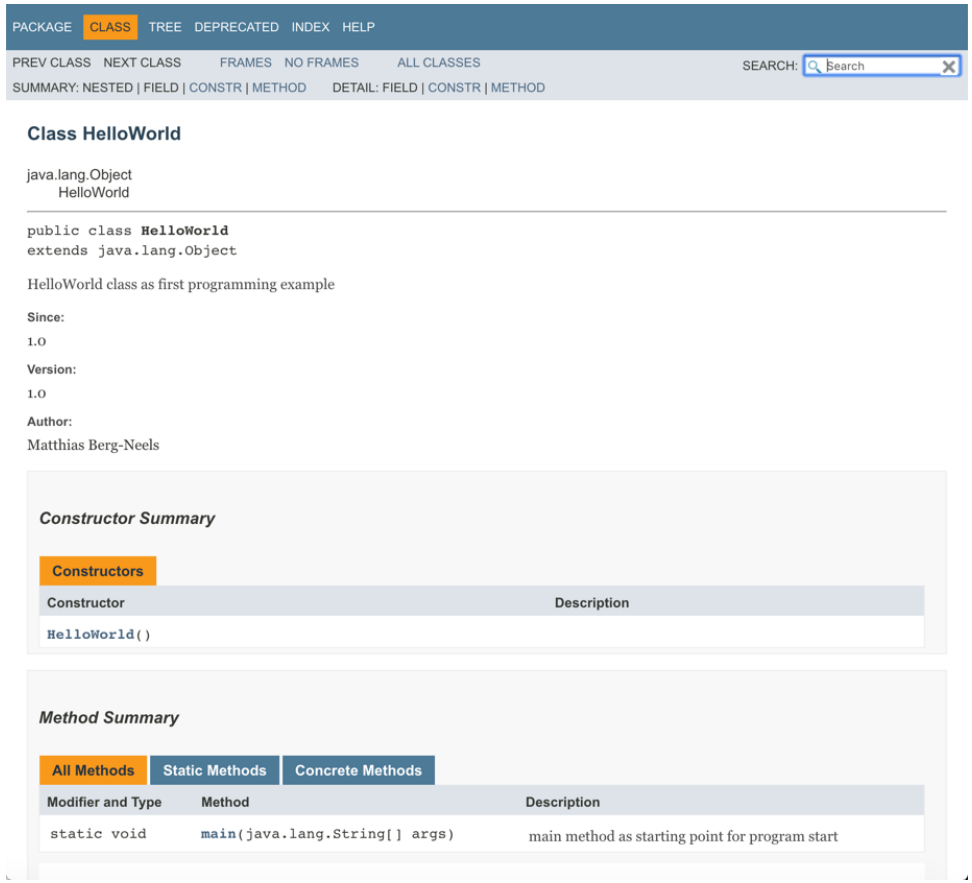
Beispiel:

```
/** HelloWorld class as first programming example
* @author Matthias Berg-Neels
* @version 1.0
*/
class HelloWorld {

  /** main method as starting point for program start
  * @param args String array for parameters from the console
  */
  public static void main (String[] args){

    System.out.println("Hello to the Java World");
  }
}
```

GENERIERUNG DER DOKUMENTATION

```
javadoc HelloWorld.java -html5 -author -version
```

Ergebnis:



PACKAGE CLASS TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES SEARCH: Search

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class HelloWorld

java.lang.Object
HelloWorld

public class **HelloWorld**
extends java.lang.Object

HelloWorld class as first programming example

Since:

1.0

Version:

1.0

Author:

Matthias Berg-Neels

Constructor Summary

Constructors

Constructor	Description
HelloWorld()	

Method Summary

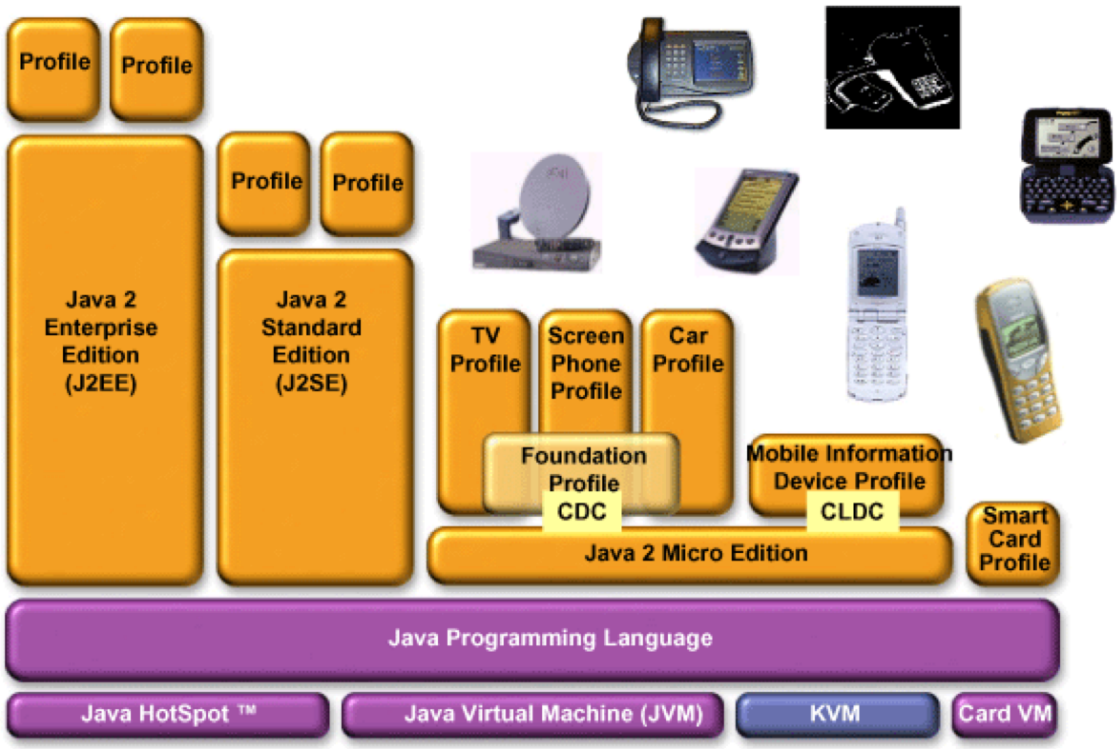
All Methods | Static Methods | Concrete Methods

Modifier and Type	Method	Description
static void	main(java.lang.String[] args)	main method as starting point for program start

DAS PAKETKONZEPT IN JAVA

- Möglichkeit zur Strukturierung bzw. sinnvollenSortierung von Klassen
- Pakete sind somit Sammlungen von Klassen
- die Klassen verfolgen einen gemeinsamen Zweck
- jede Klasse ist genau einem Paket zugeordnet
- Paketnamen können aus mehreren Teilen bestehen und hierarchisch aufgebaut sein (vergleichbar mit der Ordnerstruktur im Windows-Explorer)
- eine Klasse kann eindeutig über das Paket und ihren Namen identifiziert werden

FUNKTIONSUMFANG DER JAVA 2™ PLATFORM



FUNKTIONSUMFANG DER JAVA 2™ PLATFORM

- Java 2 Standard Edition beinhaltet das Software Development Kit mit der Standard API zur Entwicklung von Java-Applikationen
- Java 2 Enterprise Edition umfasst neben der J2SE weitere Packages zur serverseitigen Programmierung (Enterprice Java Beanss, Servlets, JSP, Java-Mail-API, etc.)
- Java 2 Micro Edition stellt eine funktional kleinere Laufzeitumgebung für mobile Endgeräte (PDAs, Handys, Navigationssysteme, etc.) dar
- Connected Device Configuration (CDC) beinhaltet die komplette JVM und ist in mobilen Systemen integriert
- Connected Limited Device Configurations (CLDC) ist eine J2ME-Bibliothek zur Abdeckung gerätespezifischer Funktionen; ermöglicht die Zusammenarbeit verschiedener Geräte der gleichen Kategorie
- KVM ist die kleinste Laufzeitumgebung und wird für den Einsatz auf Geräten mit beschränkter Speicherkapazität und CPU-Leistung verwendet
- Java Card APIs definieren eine minimale Laufzeitumgebung auf SmartCards

ENTWICKLUNGSUMGEBUNG

INTELLIJ IDEA

- Entwickler: Jetbrains
- Integrated Development Environment (IDE) für unterschiedliche Programmierspachen und Umgebungen
  - **Java**, Javascript (Node.js), Android, Kotlin, ...
- freie Community Edition
- seit Version 9.0: kostenpflichtige Version (sehr beliebt)
- seit Version 14.0: Decompiler für Java Bytecode
- Erweiterbarkeit durch Plug-Ins
- verschiedene Ableger für weitere Programmiersprachen
  - z.B. Webstorm für PHP