Java für C++-Programmierer

Alexander Bernauer
bernauer@inf.ethz.ch

Einführung in die Übungen zu
Informatik II (D-ITET)
FS2010
ETH Zürich

Ziel

- Allgemeiner Überblick
- Kennenlernen der Suchbegriffe

Warum Java? (Marketing)

- Objektorientiert
- Erheblich einfacher als C++
- Typsicher
- Virtuelle Maschine
  - compile once, run everywhere
- Umfangreiches Ökosystem

Überblick

- Theorie
  - Die Java-Umgebung
  - Die Java-Welt
  - Die Java-Sprache
- Praxis
  - Übungsblatt 0

Die Java-Umgebung

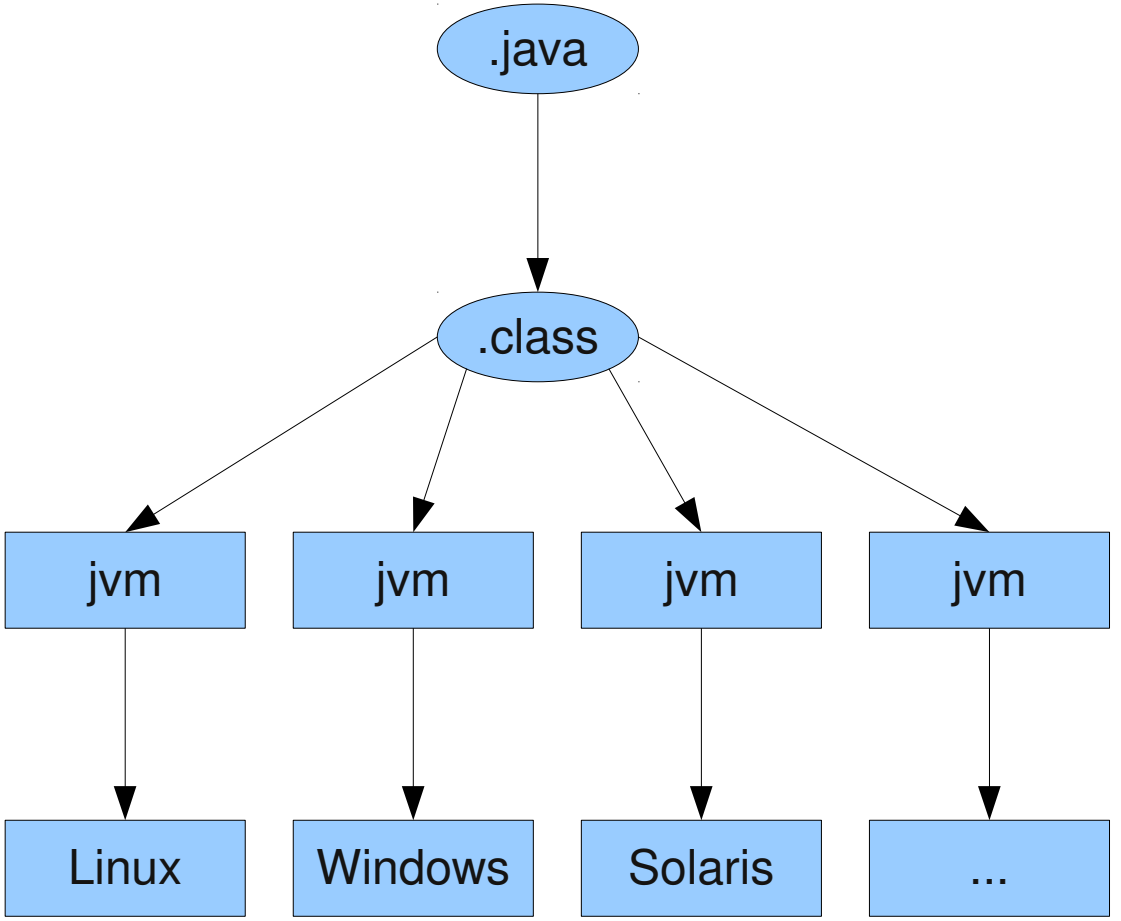
Abkürzungen

- JDK – Java Developer Kit
- SDK – Software Development Kit
- JRE – Java Runtime Environment
- JVM – Java Virtual Machine
- J2SE – Java 2 Platform, Standard Edition
- J2EE – Java 2 Platform, Enterprise Edition
- J2ME – Java 2 Platform, Micro Edition

Dateien

- .java – Quellcode
- .class – Bytecode
- .jar – Bibliothek

Virtuelle Maschine



Werkzeuge

- javac – der Compiler
- java – die virtuelle Maschine
- javap – der Disassembler
- jar – das Archivierungswerkzeug

Die Java-Welt

Ökosystem

- Umfangreiche Standardbibliothek
  - Algorithmen, Datenstrukturen, XML, Kryptographie, Kommunikation, verteilte Systeme, graphische Benutzerschnittstellen, ...
- Weitere Bibliotheken
  - Datenbanken, Parser, Web, Graphik
- Bekannte Projekte
  - Eclipse, TomCat, Ant, Hybernate

Integrierte Entwicklungsumgebung

- Editor
- Verwaltung
- Compiler
- Debugger
- Versionskontrolle
- Build System
- ...
- z.B. Eclipse, IntelliJIDEA, BlueJ, Jbuilder, NetBeans …

Javadoc

- Dokumentation durch strukturierte Kommentare
- Umwandlung in verschiedene Ausgabeformate

Unit Testing (JUnit)

- Automatisiertes Testen
- Generierung von Testberichten

Java-Plattform

- Programmiersprachen
  - Scala, Groovy, Clojure, …
- Compiler
  - OpenJDK, GCJ, ECJ
- Virtuelle Maschinen
  - Java HotSpot, Jikes, Dalvik, Kaffee, …
- Bytecodetransformatoren
- ...

Die Java-Sprache

Versionen

- JDK 1.0 – 1996
- JDK 1.1 – 1997
- J2SE 1.2 – 1998
- J2SE 1.3 – 2000
- J2SE 1.4 – 2002
- J2SE 5.0 – 2004
- Java SE 6 – 2006
- Java SE 7 - ???

Pakete

- Vermeidung von Namenskollisionen
- Der Code wird kompakter
- Pakete bilden eine Hierarchie
  - Vermittelt Kontext
  - Weltweite Eindeutigkeit
- Definition durch `package`
- Navigation mittels '.'

Organisation

- Pro öffentliche Klasse oder Schnittstelle eine gleichnamige `.java`-Datei
- Pro Klasse wird eine `.class`-Datei generiert
- Die Pakethierarchie wird auf den Verzeichnisbaum abgebildet

Bibliotheken

- Sammeln von Paketen in `.jar`-Dateien
- Distribution von `.jar`-Dateien
- Suchpfade für `.jar`-Dateien: `classpath`

Standardbibliothek

- Steht automatisch zur Verfügung
- http://java.sun.com/javase/6/docs/api/

Zugriff

- Nennung des vollständigen Namens
- Bekanntmachung eines Namens: `import`
- Die Namen aus `java.lang` sind immer bekannt
- Die Namen aus dem eigenen Paket sind auch immer bekannt

Zugriffsrechte

- `private`
- `public`
- `protected`
- `package` (Vorgabe)
- keine `friend`s
- Deklaration pro Member und nicht blockweise

Objekte

- Instanzen von Klassen
- Zugriff ausschliesslich über Referenzen
- Erzeugung mit `new`
- Entfernung durch den Garbage Collector
- kein `delete`

Primitive Typen

- `byte`, `short`, `int`, `long`
- `float`, `double`
- `boolean`
- `char`
- Können auf dem Stack angelegt werden
- Instanzen sind keine Objekte
- Korrespondierende Klassen wie z.B `Integer`

Vererbung

- Keine Mehrfachvererbung
- Statt dessen Schnittstellen: `interface`
- Alle Funktionen sind virtuell (kein `virtual`)
- Funktionen und Klassen können `abstract` sein
- Von `final` Klassen kann nicht weiter abgeleitet werden
- `final` Funktionen können in abgeleiteten Klassen nicht überschrieben werden

static

- Nur eine Bedeutung von `static`
- Methode bzw. Objekt gehört zur Klasse und nicht zur Instanz
- Navigation mittels '.'

Fehlerbehandlung

- `Error` und `Exception`
- Auslössen mit `throw`
- Behandeln mit `try/catch`
- Methoden müssen nicht behandelte Ausnahmen deklarieren: `throws`
  - ausgenommen `RuntimeException`
- Nicht behandelte Fehler führen zum Programmabbruch

Stack Traces

```
java.lang.ArrayIndexOutOfBoundsException: 3
  at example.common.TestTry.execute(Testtry.java:17)
  at example.common.TestTry.main(Testtry.java:11)
```

weitere Features

- Reflection
- Generics
- Annotations