BERUFSAKADEMIE SACHSEN
STAATLICHE STUDIENAKADEMIE
DRESDEN
UNIVERSITY OF COOPERATIVE EDUCATION

Objektorientierte Programmierung

Organisatorisches, Einführung, Geschichte

Prof. Dr.-Ing. Tenshi Hara
tenshi.hara@ba-sachsen.de

Allgemeines

- Konzepte und Strategien der objektorientierten Programmierung und geeigneter Entwicklungsumgebungen

- Lehrveranstaltung umfasst
  - Termine im Umfang von 90 Zeitstunden bestehend aus
    - Vorlesungen (OOP, Linux)
    - Diskussionen/Übungen
    - Projektarbeit in Präsenz
  - 90 Zeitstunden Heimarbeit (Projektarbeit und Nachbereitung)
- Rückmeldungen/Korrekturen/Themenwünsche bitte jederzeit

- Material
  - OPAL (3IM-OOP-20)
  - https://lern.es/oop/ (Nutzername: `student`; Kennwort: `Kaffee`)

Ziele der Lehrveranstaltung

Vermittlung von Grundlagen…

...zu Objekten, Klassen, Methoden

...zur objektorientierten Programmentwurf und -programmierung

...zu Entwicklungsumgebungen für die objektorientierte Entwicklung

Sie lernen weder effizientes Programmieren noch den detaillierten Umgang mit Entwicklungsumgebungen!

→ Es geht eher um Prinzipien und Vorgehensweisen, die am Beispiel Java und Android ausprobiert werden.

Geschichte

Java-Geschichte

- Team um James Gosling (Autor des UNIX-emacs)
- Sprache zur Entwicklung von Software für Konsumgüter
- Dafür sind „Hochsprachen“ (C/C++, usw. ungeeignet)
- Deshalb seit 1990 Entwicklung von Java
- Angelehnt an C/C++
- James Gosling wählte den Namen „Oak“ (Object Application Kernel)
- „Oak“ (Eiche) gab es schon, deshalb „Java“
- „JavaSoft“ ist Tochterfirma von Sun Microsystems
- 2010 wurde Sun Microsystems von Oracle übernommen

Das Internet und Java

- anfangs nur Daten- und Dokumentenaustausch
- 1989 Tim Berners-Lee (Physiker am CERN) Konzept des WWW
- 1992 wurde WWW international bekannt
- erste Web-Browser („Mosaic“) durch NCSA (National Center for Supercomputing Applications)
- WebRunner: Web-Browser des Java-Teams (vollständig in Java geschrieben)
- Umbenennung von Webrunner in HotJava durch Urheberrechte
- HotJava demonstrierte Bedeutung von Java für Internet
- Mai 1995 SunWorld-Konferenz (San Francisco) Publizierung der Java-Technologie
- Netscape soll Java unterstützen

Entwicklungsstufen von Java (1/6)

- JDK 1.0 (Java Development Kit 1.0) im Jahr 1996
  - Grundlegende Objekte (Objekte, Strings, Threads, Ein- und Ausgabe, Datenstrukturen, Systemeigenschaften)
  - Grafikprogrammierung (AWT – Abstract Window Toolkit)
  - Netzwerkbetrieb (URLs, TCP- und UDP-Sockets)
- JDK 1.1 im Jahr 1997
  - Vereinfachung von GUI-Programmierung (Eventhandling)
  - Internationalisierung
  - Sicherheit (verschiedene Stufen realisierbar)
  - JavaBeans, JAR-Format, Datenbankabbindung
  - RMI (Remote Method Invocation), Reflections
  - JNI (Java Native Interface)
  - mathematische Hilfsklassen (`java.math`)

Entwicklungsstufen von Java (2/6)

- JDK 1.2.x (Java 2) im Jahr 1998
  - Sicherheitskonzept (Zertifizierung)
  - JFC (Java Foundation Classes)
  - Java-swing, Java 2D, Drag and Drop
  - leichtere Interaktion von Java-Programmen mit Außenwelt
  - Collections-API (Java Collections Applications Programming Interface) meist Daten-Container
  - IDL (Java Interface Definition Language) für CORBA (Common Object Request Broker Architecture)
  - weitere Erweiterungen (Audio, Versionsidentifikation, usw.)

Entwicklungsstufen von Java (3/6)

- JDK 1.3.x „Kestrel“ Jahr 2000
  - Einführung der HotSpot-Technologie
  - Java-Plug-In steht unter Windows über die Systemsteuerung zur Verfügung
- JDK 1.4.x Jahr 2002
  - keine klassische JVM (ohne Hotspot-Optimierung) mehr
  - Unterstützung von Unicode der Version 3.0
  - weitere neue I/O APIs
  - Java-Plug-In 1.4 u.a. mit Multiversionssupport
  - Security Features JCE, JSSE und JAAS
  - Web-Start-Technologie
  - XML

Entwicklungsstufen von Java (4/6)

- Version 5.0 der Java 2 Standard Edition („Tiger“)
  - Autoboxing/Autounboxing (zwischen primitiven/elementaren Datentypen und Wrapperklassen)
  - Erweiterung der `for`-Schleife (`foreach` – kein Schlüsselwort)
  - variable Parameterlisten
  - statische Imports (`static import`)
  - Aufzählungstypen (`enum`)
  - Ausgabeformatierung (`out.format(…)`)
  - generische/typisierte Klassen und Connections

Entwicklungsstufen von Java (5/6)

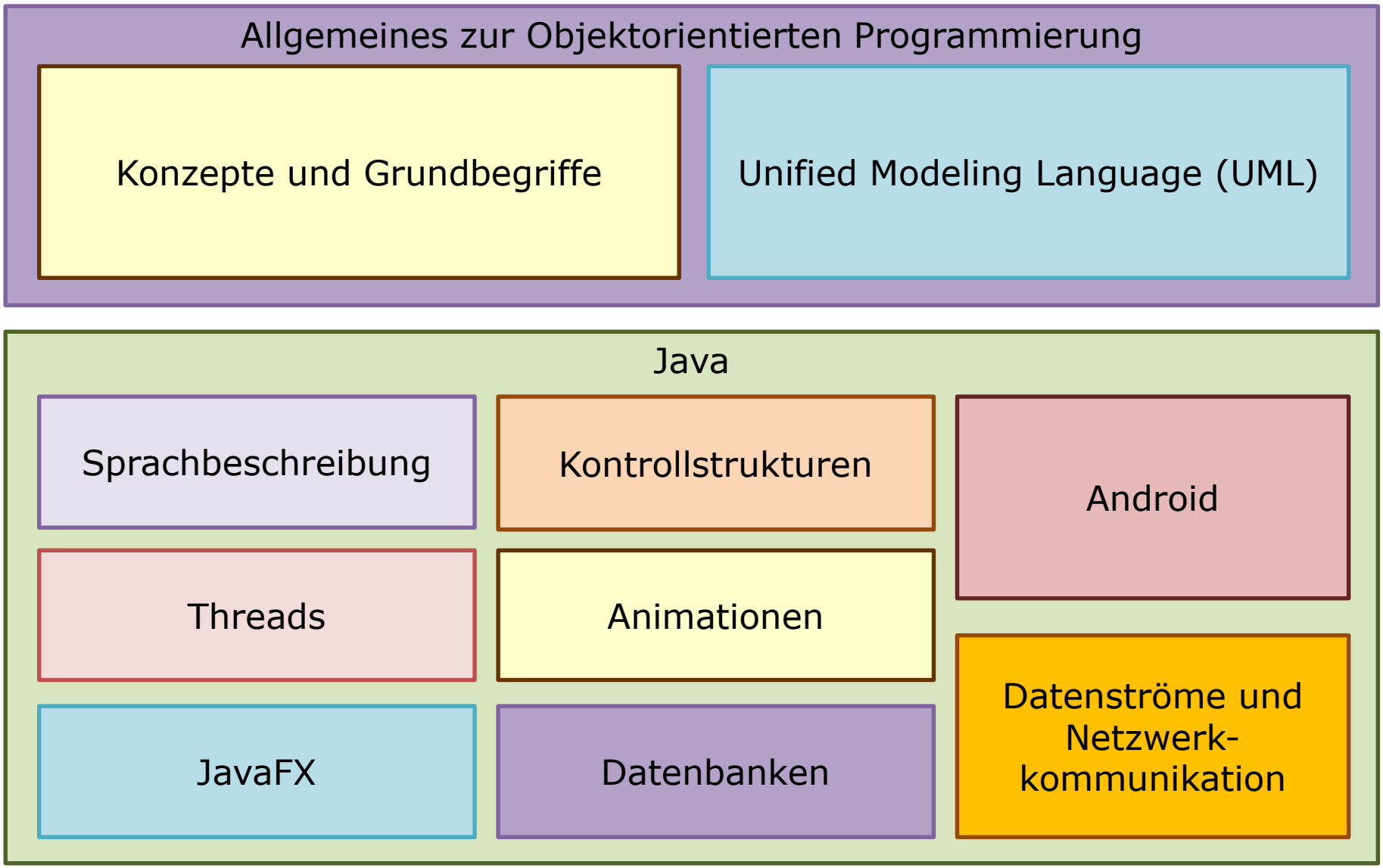
- JSE 7 – Dolphin
  - Neue Filesystem-API (NIO.2)
  - Netzwerkprotokolle (IPv6)
  - Unicode-Update (Unicode 6.0)
  - Look-and-Feel Nimbus
  - String in switch-Anweisung
  - JavaFX
  - …
- JSE 8 (älteste aktuelle LTS-Version)
  - Neue Date- und Time-API
  - Performance-Verbesserungen (Parallelisierung)
  - Lambda-Ausdrücke
  - …

Entwicklungsstufen von Java (6/6)

- JSE 9 – Dolphin
  - Erweiterung von Java um Module (Modulsystem)
  - Integration der jshell
  - grundlegende Umstrukturierung des Java-Plattform
- JSE 10
  - beständige Speicherfreigabe (Perennial Garbage Collection)
  - neues Schlüsselwort `var` für einfachere Variablenerzeugung und verbesserte Quellkodelesbarkeit
- seit JSE 11 (mit JSE 17 aktuelle LTS-Version)
  - Einführung eines kostenpflichtigen Abonnementmodells
  - nur Entwicklungs- und Testumgebung bleibt (vorerst) kostenfrei
- aktuelle Version: JSE 18

Aufbau der Lehrveranstaltung

Aufbau der Lehrveranstaltung



Übungs-/Laboreinheiten

- Sie sind selbst für Ihre Übungen zur OOP verantwortlich
  - orientieren Sie sich z.B. am LearnJ-Tutorial:
    https://www.learnj.de/doku.php
  - oder am Java-Tutorial des W3C:
    https://www.w3schools.com/java/exercise.asp
    https://www.w3resource.com/java-exercises/
  - achten Sie darauf, dass Sie die Konzepte der Vorlesung wiederholen, anwenden und verstetigen
- in den Übungs-/Laboreinheiten sollen Sie zügig dazu übergehen, an Ihrer Projektaufgabe zu arbeiten (Team-Orga, Framework-Nutzung, etc.)
  → Übungsaufgaben können und sollen aber auch diskutiert werden

Literatur der Lehrveranstaltungsreihe

Java 2 SDK v 1.2.2, Grundlagen Programmierung, HERDT-Verlag für Bildungsmedien GmbH, Nackenheim

Krüger, G.: Handbuch der Java-Programmierung, Addison-Wesley Verlag, ISBN 3-8273-2201-4

Schröter, B.: KompaktReferenz Java2, DATA BECKER GmbH & Co. KG, ISBN 3-8158-1477-4

Object Management Group (OMG) (Hrsg.): Unified Modeling Language (UML) 2.5.1 Specification

Gosling, J.; Joy B.; Steele, G.; Bracha, G.; Buckley, A.; Smith, D.: The Java® Language Specification

Ullenboom, C.: Java ist auch eine Insel: Programmieren lernen mit dem Standardwerk für Java-Entwickler, aktuell zu Java 8. (Galileo Computing)

Braun, R.; Esswein, W.; Greiffenberg, S.: Einführung in die Programmierung: Grundlagen, Java, UML (Springer-Lehrbuch)

Java Programming Basics bei https://www.udacity.com

Android Development for Beginners bei https://www.udacity.com

weitere Internet-Seiten (Wikipedia, …)

Prüfungsleistung

Prüfungsleistung

zwei Teilprüfungen

- OOP: Programmentwurf als Gruppenprüfungsleistung
  - komplett in der Theoriephase
  - Abgabe am Ende der Theoriephase
  - Beachten Sie die Hinweise im separaten Hinweisdokument!
- Entwicklungsumgebungen: Klausur zu Linux
  - 120 Minuten am Ende des 2. Theoriesemesters

Bearbeitung der Aufgabe

- Bildung von 5er-Teams in den ersten Wochen
- Vorschlag für Programmieraufgabe spätestens am 6. Mai 2022
- eigenständige Bearbeitung eines Softwareprojektes im 5er-Team
- Abgabe eines Fortschrittsberichts am 10. Juni 2022
- Abgabe der Unterlagen spätestens am 15. Juli 2022

Die Fristen sind fest. Verspätungen führen zur Note 5,0!

Im realen Projekt können Sie den Kunden auch nicht ohne Strafe auf spätere Termine vertrösten!

Rahmenbedingungen

- frei wählbar, kann auch Aufgabe des Praxispartners sein
- frei wählbare Entwicklungsumgebung
  - im Rahmen der Software in den Laboren der SSA Dresden oder
  - auf eigenen Rechnern installierte Programme
- Dokumentationswerkzeuge der Entwicklungsumgebung (bspw. Java-Doc) müssen verwendet werden
- selbstverständlich darf und soll auch außerhalb der geplanten Übungsstunden an der Aufgabe gearbeitet werden

Abgabe eines Vorschlags

- Titel der Aufgabe
- Namen der Beteiligen
- kurze verbale Erläuterung der Aufgabe in einigen Zeilen stichpunktartig
- (Hand-)Skizze zur Vermittlung einer Grobvorstellung → Implementation kann von Skizze abweichen
- geplanter Funktionsumfang (stichpunktartig)
- Erweiterungen, Verbesserungen für eventuell spätere Fortsetzung des Projektes

Programmieraufgabe

Namen und Seminargruppe
der Studierenden

Aufgabe: Das Konzept zu unserem Programm kommt aus der App-Kategorie „Real Life Achievements". Die Idee dahinter ist es, Aufgaben zu erfüllen, für die man eine bestimmte Anzahl Erfahrungspunkte erhält. Diese Punkte schalten Belohnungen frei und bestimmen den Level des Benutzers. Die Aufgaben werden nicht innerhalb der App sondern im wahren Leben gelöst, zum Beispiel: Müll runterbringen, mehr Wasser trinken, Sport machen etc.
Begleiten und motivieren soll eine Art Maskottchen.

Skizzen:

Mindestumfang: Umsetzung als Android-App. Der Benutzer kann aus Aufgaben wählen, eigene Aufgaben erstellen, löschen und als erledigt markieren. Aufgaben sind in Kategorien wie Gesundheit, Schule/Beruf, Ernährung etc. unterteilt.
Bei Abschluss einer Aufgabe erhält man Punkte, die auf dem Benutzerkonto gespeichert werden. Ab bestimmten Punktzahlen erhält man Belohnungen und steigt im Level auf.

Mögliche Erweiterungen:

- Einführung/ Anleitung beim 1. Starten
- Erstellung eigener Kategorien
- Mini Games mit dem Charakter
- Auswahl aus verschiedenen Charakteren
- Verändern des Aussehens bei Level-Aufstieg
- Veränderung des Aussehens durch Ausrüstung/Kleidung aus Belohnungen
- Kalenderansicht
- Erinnerungen
- Statusleisten/ Ladebalken für größere Aufgaben
- Teilen in der Community

Dresden, 17.05.17 Unterschriften

Datum, Unterschriften

Ablauf der Aufgabe als Programmierprojekt

- Entwicklungsumgebung festlegen
- Software-Mittel zur Umsetzung festlegen (Unity, Swing, .NET, …)
- Programmierrichtlinien und/oder Kode-Konventionen festlegen (verbindliche Festlegungen zu Namensgebungen, Formatierungen, …)
- Art der Software-Dokumentation festlegen (JavaDoc, UML, …)
- Projekt-Dokumentation vom ersten Tag an führen

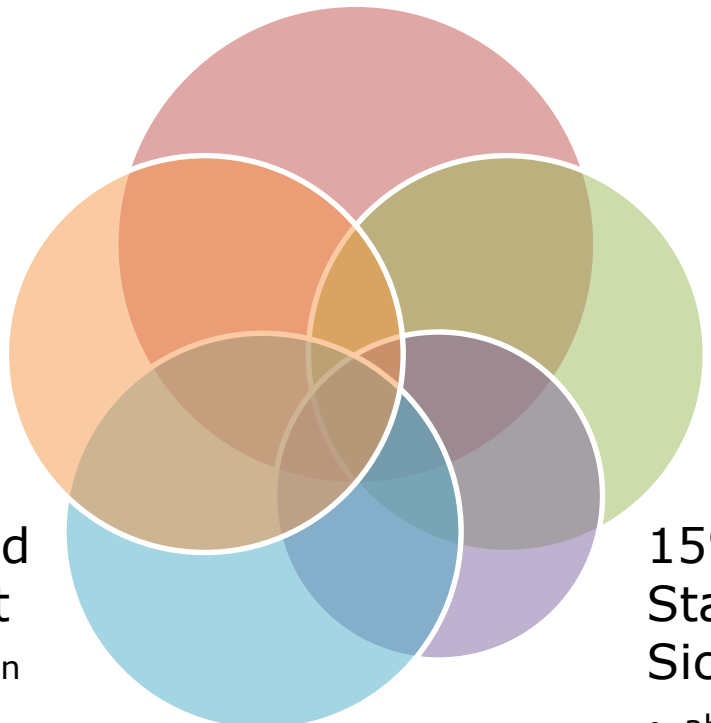
Bewertung

20% Funktionalität

- „geschlossene“ Software
- lauffähige Software

35% Dokumentation

- Nachvollziehbarkeit der Arbeitslastverteilung
- Vorgehensentscheidungen
- Team-Einteilung (Aufgaben)
- …



15% Design und Bedienbarkeit

- einfach startbar
- Info
- Hilfe
- …

15% Wartbarkeit und Erweiterbarkeit

- Quellkode vorhanden
- Einhaltung der Programmierrichtlinien
- UML-Diagramme
- (Inline-)Dokumentation

15% Stabilität und Sicherheit

- absturzsicher
- Abfangen von Fehlern
- sinnvolle Fehlermeldungen
- …