Teaching Object-Oriented Programming with the Objects-first Approach
An Experience Report
SEUH 2023
© Sandro Speth
Motivation

Heterogeneous group of students*

* In “PSE”: ~ 800 students & > 20 study subjects
Motivation

Bottom-up

Objects-first
Course Structure

- Lecture
  - 2x 90 minutes, weekly
  - 21 lecture units (1-2 lecture slots each)

- Exercise
  - 1x 90 minutes, weekly
  - 13 exercise sheets

- Lecture Exercise
  - 1x 90 minutes, weekly
  - Repetition of lecture content and discussions
Objects-first with Mini Programming Worlds

```java
import de.hamstersimulator.objectsfirst.datatypes.Direction;
import de.hamstersimulator.objectsfirst.datatypes.Location;
import de.hamstersimulator.objects.first.model.hamster;
import de.unistuttgart.iste.sqa.pse.sheet5.presence.controlflowexercise.BaseControlFlowHamsterGame;

import java.util.Optional;
import java.util.concurrent.ThreadLocalRandom;

public class EmptyMouthGame extends BaseControlFlowHamsterGame {
    public EmptyMouthGame() {
        super("/territories/territoryExample05-7.ter");
    }

    /**
     * Starts the game with a predetermined territory and lets Paule walk through a short test scenario.
     * Do not modify!
     */
    @Override
    protected void run() {
        final Hamster stefen = new Hamster(this.game.getTerritory(), new Location(2, 4), Direction.SOUTH, new GrainCounter(3));

        while (paule.grainAvailable()) {
            paule.pickGrain();
        }

        for (int i = 0; i < 11; i++) {
            paule.move();
        }

        EmptyMouth();

        stefen.pickGrain() : void;
        paule.pickGrain() : void;
    }  
```

Sandro Speth - Teaching Object-Oriented Programming with the Objects-first Approach: An Experience Report
Content of the Lecture

Auffällig ist der starke Kontrast der praktischen Anteile (wirklich alle in derselben Miniwelt?) und dem hohen theoretischen Anspruch der Vorlesung (u.a. mit den Themen abstrakte Syntaxbäume (in Woche 2!), Korrektheit, Invarianten, Vertragsmodell, multiple Vererbung, bis hin zu RE und Softwarearchitektur wird ein extrem ambitionierter Bogen geschlagen).

- Reviewer 2
Content of the Lecture

- Objects & Class Foundation and Usage
- Data Types, Variables, and Control Flow
- Own Java Classes, and complex OOP
- Software Engineering Basics
Content of the Lecture

- Motivation & Intro to CS + Elementary CS concepts
- Basic understanding of programs with Hamster Simulator
  - Objects which call operations
  - Command vs. Query
  - Building stories through sequence of interactions
- Structure of programs
  - Instruction, expression, lexes, syntax, semantics, etc.
  - Simplified ASTs
- Interfaces and documentation
- Logic

= extremely difficult for students
Content of the Lecture

- Basics of Object usage
  - Instantiation, null, and this

- Control flow
  - Sequence, Loop, Error handling
  - Loop variants and invariants

- Types and variables
  - Visibility, releasing variables, read-only
  - Primitive vs. complex data types
  - Equals vs. same
  - Immutable

= extremely difficult for students
Content of the Lecture

- Programming own Java classes
  - Static vs. non-static
  - Functional decomposition
  - UAP
  - Offensive and defensive programming
- “Complex” OOP
  - Inheritance, Polymorphism
  - Abstraction and Interfaces
  - Overloading and Overriding
  - Static and dynamic types
- “Complex” OOP cont.
  - Type conformance
  - Constructor chaining
  - Diamond problem
  - Liskov’s Substitution Principle
  - Collections
- “Christmas Lecture”
  - Outline of different programming languages

= extremely difficult for students
Content of the Lecture

- Clean Code
- Recursion
- Modelling
  - Class and object diagram
  - Sequence diagram
- Various SE topics
  - Development processes
  - RE
  - SA
- Testing

- Functional programming in Java
  - Anonymous classes
  - Lambdas, method references
  - Java Streams API
- Semantic verification
  - Weakest precondition
  - GUIs, events and listeners

= extremely difficult for students

Sandro Speth - Teaching Object-Oriented Programming with the Objects-first Approach: An Experience Report
Discussion – Exam

~800 students

Written exam (~55% fail)

„Die Prüfung [...] testet somit höchstens theoretisches Wissen, keine Programmierfähigkeiten“ – Reviewer 2

Kapiervorgang abgebrochen
Discussion – Lecture & Exercise Evaluation

- Questionnaire
  - Likert Scale from 1 (highly positive/agree) to 5 (highly negative/disagree)
  - Free text answers

Lecture Survey

Exercise Survey

Lecture Exercise Survey

~2 – 2.3

~1.8 – 2

~1.3 – 1.8
Discussion – Lecture & Exercise Evaluation

Free text answers

👍 Comprehensibility

👍 Clearly well-structured content

👍 High amount of practical programming tasks

👎 Amount of content

👎 Effort in exercise sheets for students without prior knowledge

👎 Choice of programming language
Discussion – Objects-first

- No prior knowledge often better than prior knowledge
  - Less bored
  - Prior knowledge results in missing content
- MPWs, e.g., Hamster Simulator
  - Simplicity in the beginning
  - Wish for more complex MPWs later at the semester
→ Generation of MPWs in different programming languages and complexity with MDSD
Related Work

- No support for newer Java versions (>11)
- Students were fighting bugs regularly (applies to BlueJ)
- Do not offer more elaborate IDE features (good debugger, refactorings, etc.)
- We plan to support MPWs in other programming languages than Java

Schmolitzky et al. [SZ07]
- Two semesters
- Different topics & order
- Include DSA but no SE concepts

Cooper et al. [CDP03]
- 3D MPW
- Scratch-like programming
- No topics discussed
Conclusion

Objects & Class Foundation and Usage
Data Types, Variables, and Control Flow
Own Java Classes, and complex OOP
Software Engineering Basics

Lecture Survey
~2 – 2.3

Exercise Survey
~1.8 – 2

Lecture Exercise Survey
~1.3 – 1.8

Written exam
(~55% fail)
Outlook – Gamify-IT

Which Java Class implements an ordered collection that supports duplicates?

Set
List
MultiSet
Map

public static int main(String... args) {
    System.out.println("ok");
}
Outlook – Model-driven MPW Generation
Conclusion

Objects & Class Foundation and Usage

Data Types, Variables, and Control Flow

Own Java Classes, and complex OOP

Software Engineering Basics

Lecture Survey

~2 – 2.3

Exercise Survey

~1.8 – 2

Lecture Exercise Survey

~1.3 – 1.8

Written exam

(~55% fail)
Thank you!

Sandro Speth

e-mail  sandro.speth@iste.uni-stuttgart.de
phone  +49 (0) 711 685-61693
www.  iste.uni-stuttgart.de/sqa/team/Speth

University of Stuttgart
Institute of Software Engineering,
Software Quality and Architecture Group
Universitätsstraße 38,
70569 Stuttgart
Room 1.336