University of Stuttgart
Institute of Software Engineering (ISTE)
Software Quality and Architecture Group (SQA)

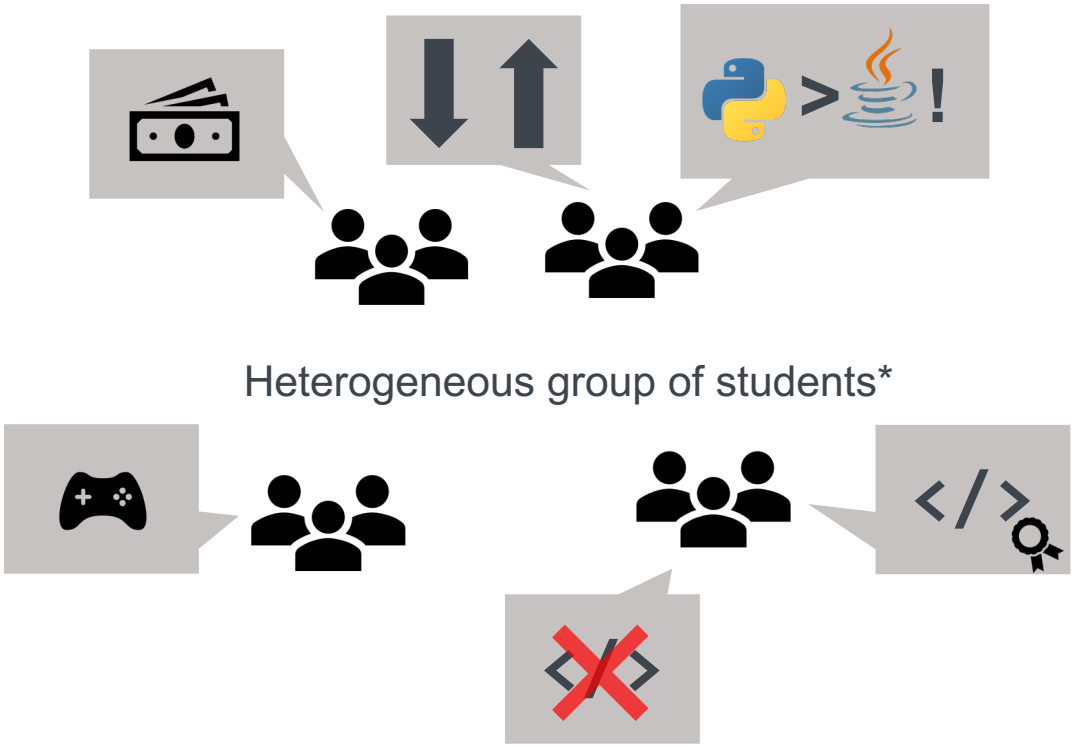
Teaching Object-Oriented Programming with the Objects-first Approach

An Experience Report

SEUH 2023

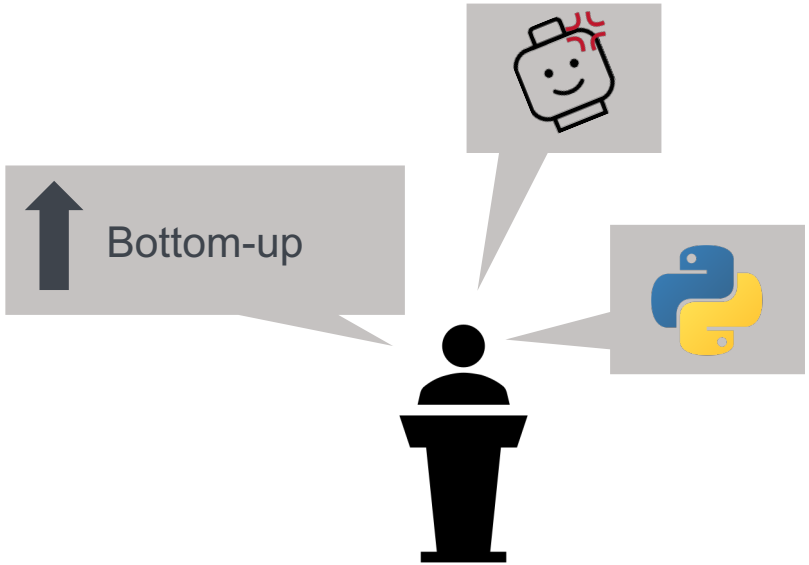
© Sandro Speth

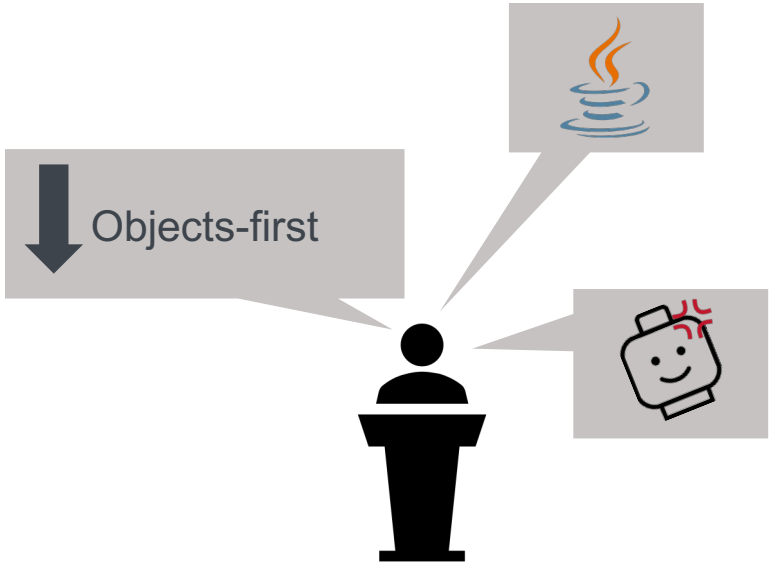
Motivation



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

Motivation





Course Structure

9 ECTS (“4+2(+2)” model)

Lecture

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

Exercise

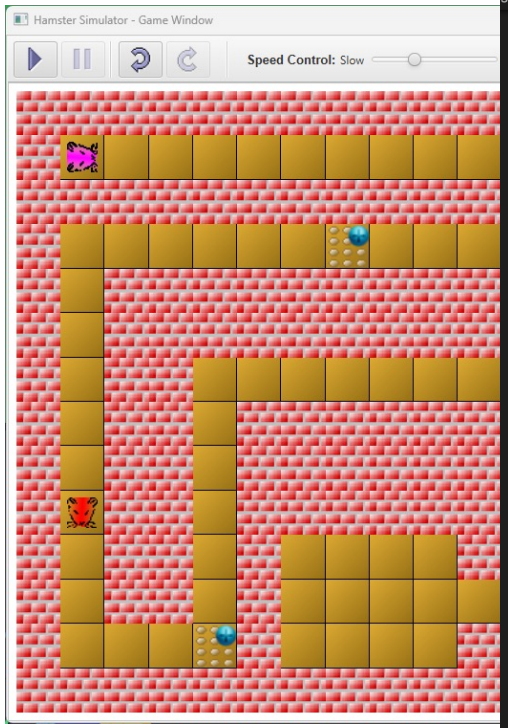
- 1x 90 minutes, weekly
- 13 exercise sheets

Lecture Exercise

- 1x 90 minutes, weekly
- Repetition of lecture content and discussions

Objects-first with Mini Programming Worlds

SQA Hamster Simulator




```
src > main > java > de > unistuttgart > iste > sqa > pse > sheet05 > presence > controlflowexercise > games > EmptyMouthGame.java > EmptyMouthGame > run()
1  package de.unistuttgart.iste.sqa.pse.sheet05.presence.controlflowexercise.games;
2
3  import de.hamstersimulator.objectsfirst.datatypes.Direction;
4  import de.hamstersimulator.objectsfirst.datatypes.Location;
5  import de.hamstersimulator.objectsfirst.external.model.Hamster;
6  import de.unistuttgart.iste.sqa.pse.sheet05.presence.controlflowexercise.BaseControlFlowHamsterGame;
7
8  import java.util.Optional;
9  import java.util.concurrent.ThreadLocalRandom;
10
   You, vor 1 Sekunde | 3 authors (st156832 and others)
11 public class EmptyMouthGame extends BaseControlFlowHamsterGame {
12
13     public EmptyMouthGame() {
14         super(territoryFile: "/territories/territoryExample05-7.ter");
15     }
16
17     /**
18      * Starts the game with a predetermined territory and lets Paule walk through a short test scenario.
19      * Do not modify!
20      */
21     @Override
22     protected void run() {
23         final Hamster steffen = new Hamster(this.game.getTerritory(), new Location(2, 4), Direction.SOUTH, newGrainCount: 3);
24
25         while (paule.grainAvailable()) {
26             paule.pickGrain();
27         }
28         for (int i = 0; i < 11; i++) {
29             paule.move();
30         }
31         emptyMouth();
32
33         steffen.p      You, jetzt • Uncommitted changes
34                 pickGrain() : void                Hamster.pickGrain() : void
35     }           putGrain() : void
36 }               prf                               Private field
```

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

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

Discussion – Exam

~800 students

Written exam
(~55% fail)

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

es

Discussion – Lecture & Exercise Evaluation

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

Lecture Survey

~2 – 2.3

Exercise Survey

~1.8 – 2

Lecture Exercise Survey

~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

 BlueJ  BlueJ

- 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

Lecture Survey

~2 – 2.3

Exercise Survey

~1.8 – 2

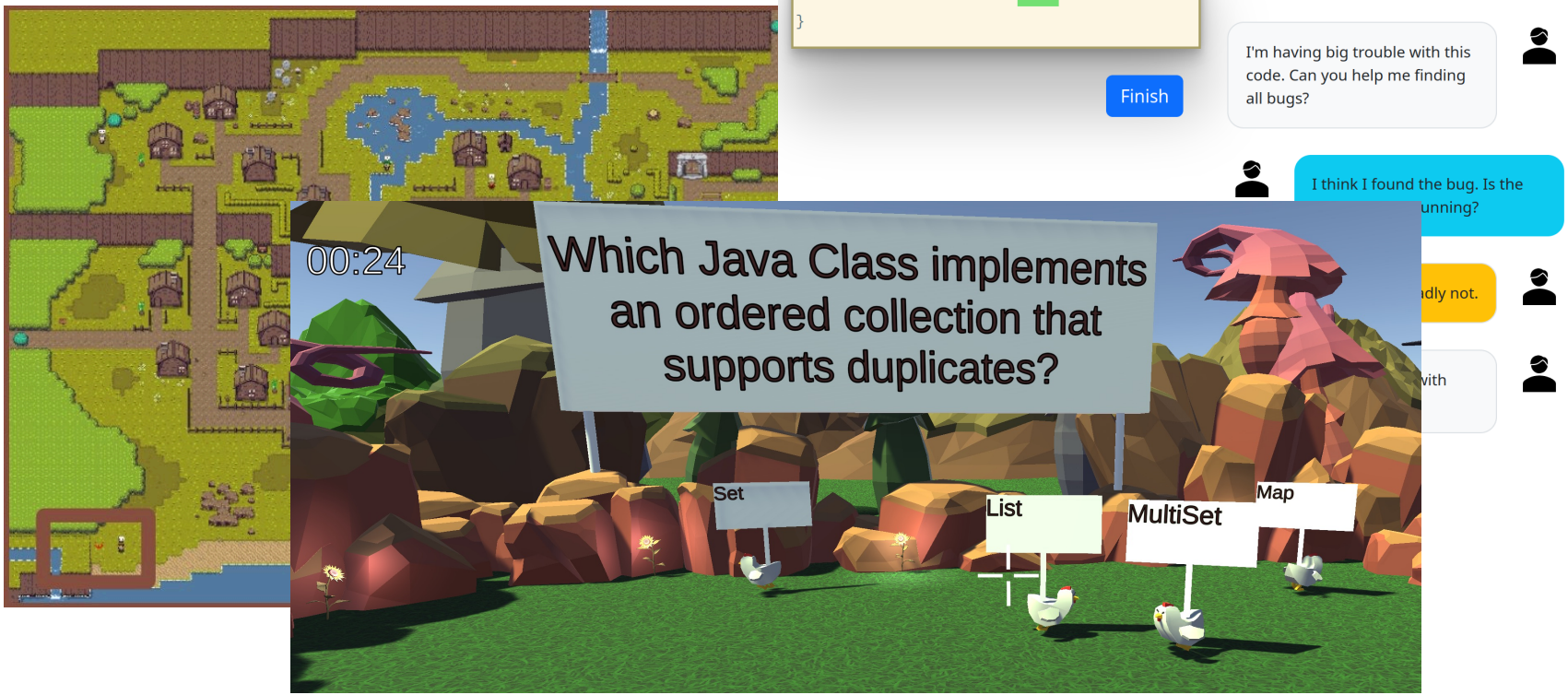
Lecture Exercise Survey

~1.3 – 1.8

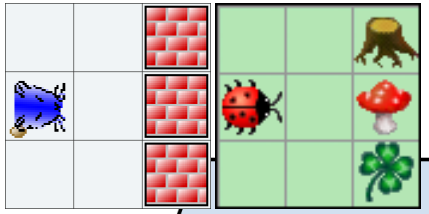
Written exam

(~55% fail)

Outlook – Gamify-IT



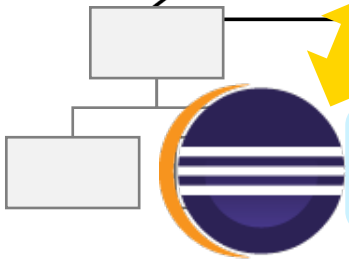
Outlook – Model-driven MPW Generation



Breadcrumb Tutorial

Entities	Queries	Transformations	Generation Templates	Target Code
Commands	Constraints	Intermediate Models		

Modeling → Transformation → Generation



Conclusion

Lecture Survey

~2 – 2.3

Exercise Survey

~1.8 – 2

Lecture Exercise Survey

~1.3 – 1.8

Written exam

(~55% fail)

University of Stuttgart
Institute of Software Engineering (ISTE)
Software Quality and Architecture Group (SQA)

@spethso

/in/sandro-speth

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