KU LEUVEN

Programmeertechnieken/Programming Techniques

Part 0: Introduction

Koen Pelsmaekers

Campus Groep T, 2022-2023

Who am I?

- Koen Pelsmaekers (oe = [u])
- @Groep T: 1986
- 60% examombuds/study track counselor/(ict)/OC chairman
- 40% programming courses
  - Long, long ago… Pascal, C, C++ … current interest: Kotlin
  - First Java course @ Groep T: 1997
  - Lab OOP, Programming Techniques, UX-driven Web Development (R&D Experience), Distributed Applications (until 2021-2022)
- Contact
  - room GT 03.14.05 or GT 01.4.02/1 (exam ombuds/study track counselor)
  - koen.pelsmaekers@kuleuven.be

Course goals: “Programming Techniques”

- Advanced object-oriented programming: polymorphism (inheritance, abstract classes, interface) and dynamic binding (aka polymorphic or runtime or late binding)
- Data structures: “beyond ArrayList”
- Functional programming (streams/lambda expressions)
- Programming language constructs
- Design patterns and refactoring (clean code)
- Concurrent programming
- Develop a “bigger” project
    - Exercises on inheritance/dynamic binding, interface
    - Android app

Prerequisite knowledge

- Courses @ Groep T
  - Objectgerichte softwareontwikkeling (T2AOS1)/Object-oriented Software Development (T2AOS2)
- Self-learning
  - Other OO-language introduction: C#, C++, Objective-C, …
  - Book: Objects First with Java (see: Courses @ Groep T)

Lecture content

- Part 0: Introduction
- Part 1: Polymorphism & dynamic binding
- Part 2: Data structures
- Part 3: Functional programming
- Part 4: Programming laguage features
- Part 5: Design patterns, refactoring & clean code
- Part 6: Introduction to concurrent programming

Course documentation & tools: lectures

- Slides (in pdf) on Toledo
- Recommended books
  - Objects First with Java. A Practical Introduction Using BlueJ. David J. Barnes, Michal Kölling.
  - Concise Guide to Object-Oriented Programming: An Accessible Approach Using Java. Kingsley Sage.
  - Clean Code: a Handbook of Agile Software Craftmanship. Robert “uncle Bob” Martin.
- Interesting books
  - Design Patterns. Elements of Reusable Object-Oriented Software. Erich Gamma et al.
  - Refactoring. Improving the Design of existing Code. Martin Fowler.
  - Concurrent Programming in Java. Design Principles and Patterns. Doug Lea.

Course documentation & tools: lab sessions

- On-line documentation (f.i. Java API, android API, …)
- Assignments on Toledo
- Software
  - Java 11 or higher (Java 17 LTS version)
  - IntelliJ IDEA (JetBrain)
    - Free student license for **Ultimate** edition
    - With Android Studio plugin or stand alone Android Studio
  - Android SDK
  - Visual Paradigm (UML diagrams)
    - Free student license (version…): see Toledo

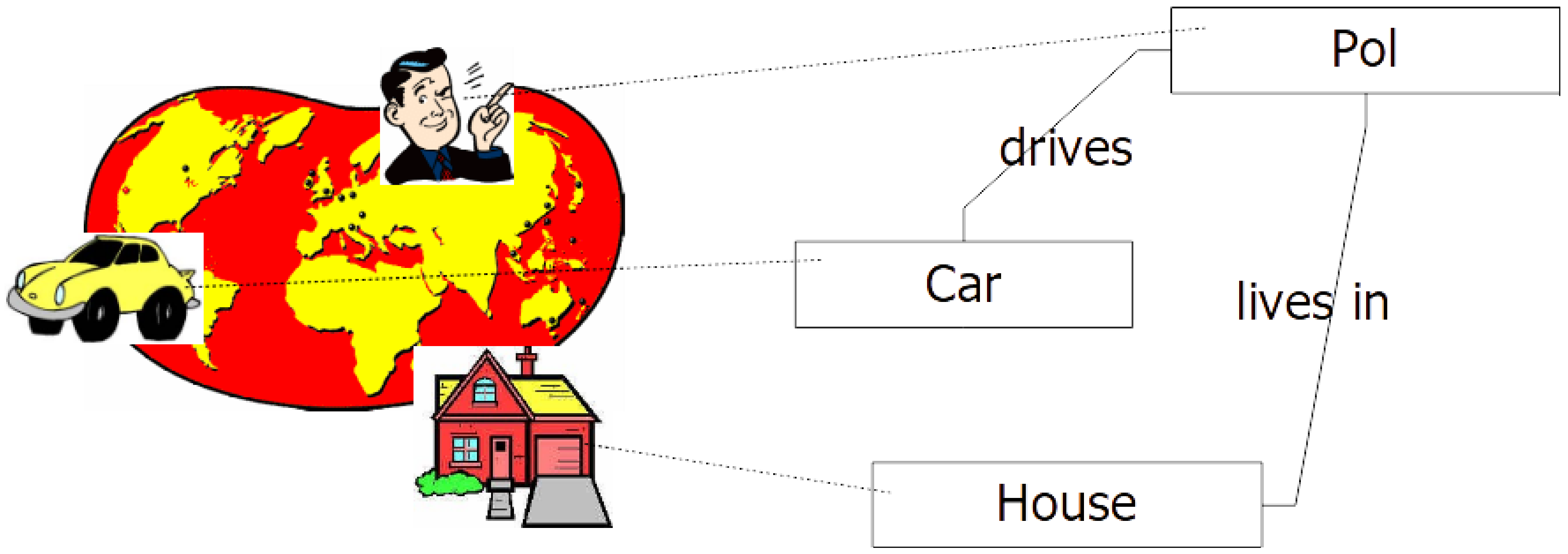
Course organisation & evaluation

- 6 credits => 6 * 25-30 hours = 150-180 hours study time
- Lectures
  - 24 hours (12 * 2 hours/week)
  - Koen Pelsmaekers
  - Evaluation: exam (written, MC and open questions) – cheat sheet of 1 A4 (both sides) <u>hand written & made by yourself</u> allowed; printed API “snippets” (aka formularium) available (weight in total result: 55%)
- Lab Sessions
  - 48 hours (12 * 4 hours/week)
  - Zhou Nianmei, Ludo Bruynseels, Stijn Langendries, Jeroen Wauters + student-assistent(s)
  - Evaluation: continuous assessment and Android project presentation (weight in total result: 45%)

Why an object-oriented approach?

- ’70: good programming == “structured programming” (vs. assembler code)
  => supporting languages: C (1972, K&R), Pascal (1970), …
- ’90: good programming == “object-oriented programming”
  => supporting languages: SmallTalk (1980), C++ (1983), Java (1995), C# (2001), Objective-C (1986), …

“Level of abstraction”


Pol
drives
Car
lives in
House

Why an object-oriented approach?

- Better modeling of the real (complex!) world
  Requirements == Analysis == Implementation
- Less and easier maintenance
  - Higher level of abstraction
    - Less sensitive to changes
    - “locality of change”
- Quality control for software
- Reusable code (long term): writing generic code is difficult
- Delay of implementation details: objects can be changed easily
- Fast release cycle