Juniorprogrammierer.de

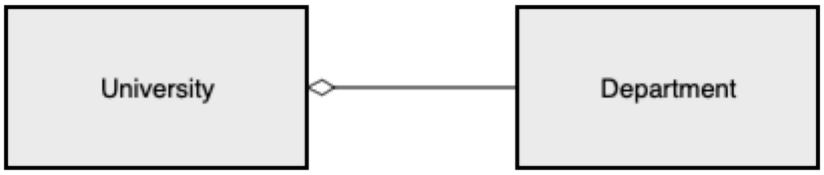
Java 12: Associationen

2024/25 – Sascha Stojanovic

Agenda

- Aggregation
- Composition
- Exercise

Aggregation



- It represents a “has-a” relationship
- Both classes can exist independently from each other
- Both are connected via a white “raute”
- Important: If 1..1/n relation put Importingpara. Into constructor, for 0..1/n just a declaration into constructor

```java
class Department {
    private String name;

    public Department(String name) {
        this.name = name;
    }

    // Other department-related methods
}

class University {
    private String name;
    private List<Department> departments; // Aggregation

    public University(String name) {
        this.name = name;
        this.departments = new ArrayList<>();
    }

    public void addDepartment(Department department) {
        departments.add(department);
    }
```

Composition

- It represents a “consist-of” or “is-part-of” relationship
- Weaker component cannot consist alone
- Both are connected via a black “raute”


```java
class Room {
    private String name;

    public Room(String name) {
        this.name = name;
    }

    // Other room-related methods
}

class House {
    private List<Room> rooms; // Composition

    public House() {
        this.rooms = new ArrayList<>();
        rooms.add(new Room("Living Room"));
        rooms.add(new Room("Bedroom"));
        rooms.add(new Room("Kitchen"));
    }

    public List<Room> getRooms() {
        return rooms;
    }
```

Exercise

- Every Vehicle has a engine (**1..1**)
- Create a Engine.java with
  - Object-variable String “engineType” and corresponding getter and setter
  - 1 parameter constructor
- Adapt the Vehicle.java and Car.java accordingly
- Now add the Rider.java
  - It has a **0..n** relationship to vehicle in form of a aggregation
  - Rider.java has only one object-var. “name” and the corresponding getter and setter
- Adapt the Vehicle.java and Car.java accordingly using a ArrayList