Informatik Q1 Abels

Beziehungen von Klassen im UML

UML

Assoziation

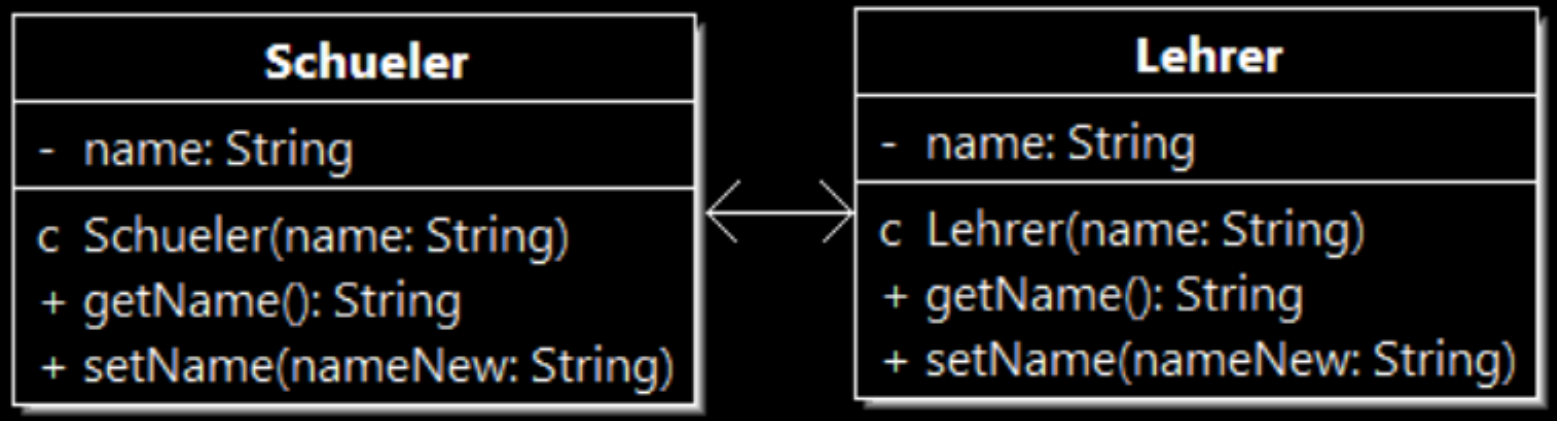
Aggregation

Komposition

Assoziation Aggregation Komposition

"kennt" – Beziehung

Eine Assoziation beschreibt die Beziehung zwischen zwei Klassen, bei der jedes Objekt einer Klasse mit Objekten einer anderen Klasse in Verbindung steht.

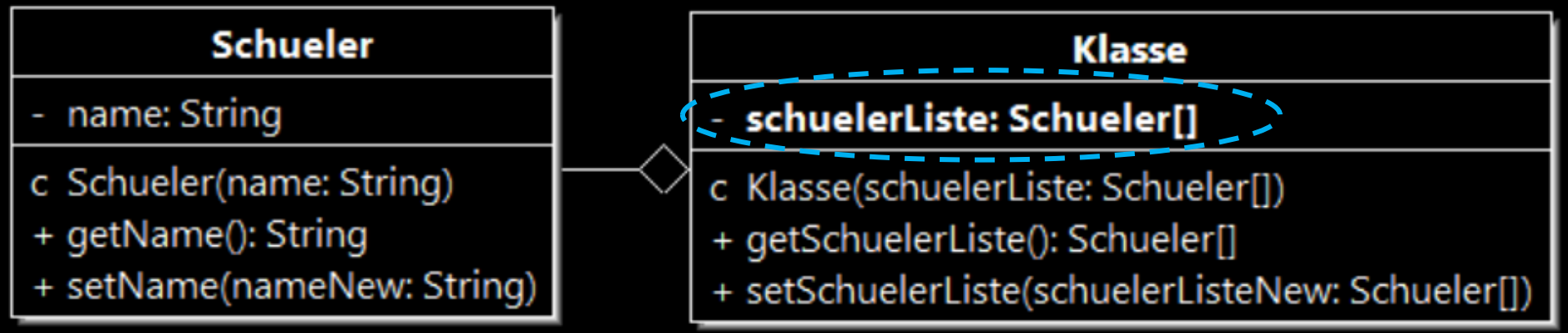


Beispiel: Die Klassen **`Schueler`** und **`Lehrer`** können in einer Assoziation stehen, da Schüler Beziehungen zu Lehrern haben. Es gibt jedoch keine direkte Abhängigkeit, und beide Klassen können unabhängig voneinander existieren.

Assoziation Aggregation Komposition

"ist-Teil-von" – Beziehung

Eine Aggregation ist eine spezielle Form der Assoziation, bei der eine Klasse (Container) eine Sammlung von Objekten einer anderen Klasse besitzt, und diese Objekte können unabhängig voneinander existieren.

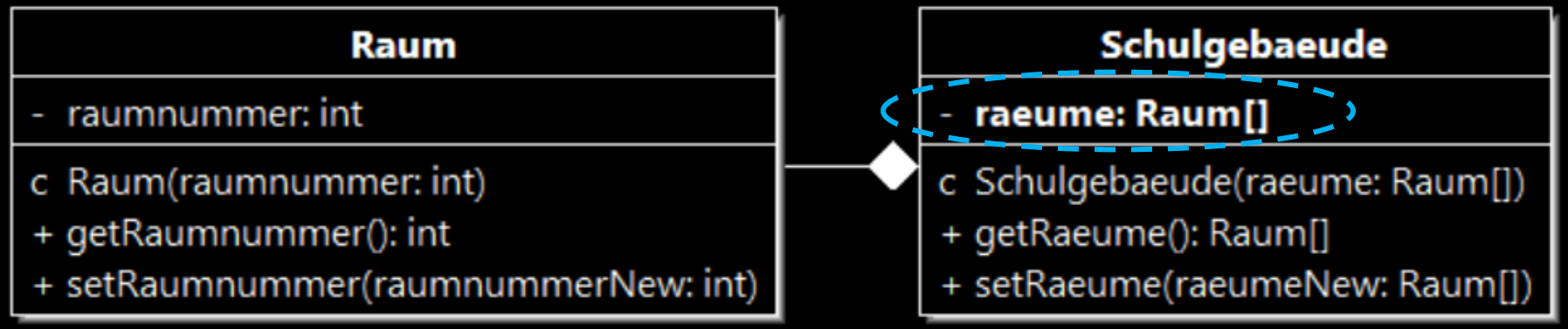


Beispiel: Die Klasse **`Klasse`** kann eine Aggregation mit der Klasse **`Schueler`** haben, da eine Klasse eine Sammlung von Schülern enthält. Die Schüler können jedoch auch ohne die Klasse existieren.

Assoziation Aggregation Komposition

"ist-Teil-von" – Beziehung

Eine Komposition ist eine strenge Form der Aggregation, bei der die Lebensdauer der enthaltenen Objekte von der Lebensdauer des übergeordneten Objekts abhängt. Wenn das übergeordnete Objekt zerstört wird, werden auch die enthaltenen Objekte zerstört.



Beispiel: Die Klasse **Raum** kann eine Komposition mit der Klasse **Schulgebaeude** haben, da der Raum nur im Kontext des Schulgebäudes existiert. Wird das Schulgebäude zerstört, endet auch die Existenz des Raums.

Übung 1

Wir betrachten einen **Buchladen**.

Die Klasse **Buch** besitzt die Attribute **Titel**, **ISBN** und **Preis**. Ein Buch wird mit seiner ISBN und seinem Titel erzeugt. Alle Attribute sollen gelesen werden können, nur der Preis soll geändert werden können. Die Klasse besitzt eine Methode **toString**, welche die Buchdaten zurückgibt.

Die Klasse **Autor** wird durch die Attribute **Name**, **Synonym** und **Gage** gekennzeichnet. Ein Autor wird mit seinem Namen erzeugt, Alle Attribute können gelesen werden, Synonym und Gage können auch gesetzt werden. Die Klasse Autor besitzt eine Methode **toString**, welche die Daten des Autors zurückgibt.

Zu jedem **Buch** soll ein **Autor** gespeichert werden.

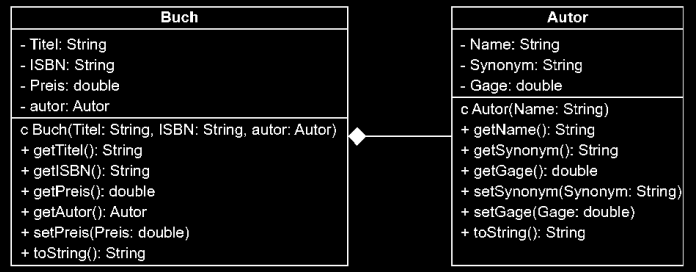
a) Modelliere ein UML-Klassendiagramm der Klassen **Buch** und **Autor** mit deren Beziehung.

b) Beschreibe, wie der **Autor** in der Klasse **Buch** durch Attribute und Methoden eingebunden wird.

c) Implementiere die Klassen **Buch** und **Autor** in Java.

d) Implementiere eine **Main**, welches drei Bücher mit ihren Autoren erzeugt und über die Methode **toString** die Daten ausgibt.

Übung 1



Buch.java

```java
package Buchladen;

public class Buch {
    private String Titel;
    private String ISBN;
    private double Preis;
    private Autor autor;

    public Buch(String Titel, String ISBN, Autor autor) {
        this.Titel = Titel;
        this.ISBN = ISBN;
        this.autor = autor;
    }

    public String getTitel() {
        return this.Titel;
    }

    public String getISBN() {
        return this.ISBN;
    }

    public double getPreis() {
        return this.Preis;
    }

    public Autor getAutor() {
        return this.autor;
    }

    public void setPreis(double Preis) {
        this.Preis = Preis;
    }

    public String toString() {
        return this.autor + ": " + this.Titel + " [" + this.ISBN + "] = " + this.Preis + " EUR";
    }
}
```

Autor.java

```java
package Buchladen;

public class Autor {
    private String Name;
    private String Synonym;
    private double Gage;

    public Autor(String Name) {
        this.Name = Name;
    }

    public String getName() {
        return this.Name;
    }

    public String getSynonym() {
        return this.Synonym;
    }

    public void setSynonym(String Synonym) {
        this.Synonym = Synonym;
    }

    public double getGage() {
        return this.Gage;
    }

    public void setGage(double Gage) {
        this.Gage = Gage;
    }

    public String toString() {
        return this.Name + " [" + this.Synonym + "] = " + this.Gage + " EUR";
    }
}
```

Main.java

```java
package Buchladen;

public class Main {
    public static void main(String[] args) {
        Autor autor1 = new Autor("Franz Kafka");
        autor1.setSynonym("Franzi");
        autor1.setGage(421.43);

        Autor autor2 = new Autor("J. R. R. Tolkien");
        autor2.setSynonym("Tolki");
        autor2.setGage(3211.32);

        Buch buch1 = new Buch("Die Verwandlung", "3709217491239", autor1);
        buch1.setPreis(42.42);

        Buch buch2 = new Buch("Der Herr der Ringe", "9948129048921", autor2);
        buch2.setPreis(98.33);

        Buch buch3 = new Buch("Das Urteil", "9912084120214", autor1);
        buch3.setPreis(12.43);

        System.out.println(buch1.toString());
        System.out.println(buch2.toString());
        System.out.println(buch3.toString());
    }
}
```

Terminal

```
Franz Kafka [Franzi] = 421.43 EUR: Die Verwandlung [3709217491239] = 42.42 EUR
J. R. R. Tolkien [Tolki] = 3211.32 EUR: Der Herr der Ringe [9948129048921] = 98.33 EUR
Franz Kafka [Franzi] = 421.43 EUR: Das Urteil [9912084120214] = 12.43 EUR
```

Übung 2

Modelliere die folgenden Situationen als UML Klassendiagramm inkl. Beziehungen.

a) Zwischen einem Musiker und einem Musik-Label besteht in der Regel eine Verbindung, die als „Vertrag“ bezeichnet wird. Musiker können bei einem oder auch keinem Label unter Vertrag stehen.

b) Ein Zug besteht aus genau 4 Waggons und einer Lok, wobei Waggons und Loks auch ohne Zug existieren können.

c) In einem Buch gibt es Seiten, Absätze und Wörter. Die Absätze können dabei über mehrere Seiten gehen. Auch ein allein stehendes Wort bildet schon einen Absatz. Jede Seite kann herausgerissen werden. Wenn das Buch verbrennt, verbrennen auch alle seine Seiten.

d) Eine Bank hat viele Kunden. Jeder Kunde besitzt einen Namen und kann über mehrere Konten verfügen. Zu jedem Konto gehören eine Kontonummer, der Kontostand und die vielen, mit dem Konto verbundenen Einzahlungen und Auszahlungen. Ein- und Auszahlungen bestehen jeweils aus einem Betrag und einem Datum

Tagebucheintrag

Assoziation

Aggregation

Komposition

Wochenübung

a) Erstelle ein UML-Klassendiagramm zum Thema **`Fitnessstudio`**.

b) Implementiere entsprechende Klassen sowie eine Hauptklasse mit Tests.