

Seite 8. UML-Klassendiagramm-Trainer

UML-Klassendiagramm-Trainer

Dieser interaktive Trainer gehört zur Theorie-Seite:

UML-Klassendiagramm

Hier übst du, UML-Klassendiagramme richtig zu lesen und aus Aufgabenstellungen abzuleiten.

Im Fokus stehen:

Klassen
Attribute
Methoden
Sichtbarkeit
Assoziation
Vererbung
Aggregation
Komposition
korrekte UML-Linienenden

Was wird trainiert?

Bereich	Bedeutung
Klassen erkennen	wichtige Programmobjekte aus der Aufgabe ableiten
Attribute zuordnen	gespeicherte Eigenschaften in die richtige Klasse schreiben
Methoden eintragen	Verhalten/Funktionen einer Klasse ergänzen
Sichtbarkeit beachten	<code>+</code> , <code>-</code> , <code>#</code> korrekt selbst eintragen
Vererbung erkennen	hohles Dreieck zeigt zur Oberklasse
Aggregation erkennen	leere Raute steht am Ganzen
Komposition erkennen	gefüllte Raute steht am Ganzen
Assoziation erkennen	einfache Linie zwischen Klassen

Interaktiver UML-Klassendiagramm-Trainer

UML-Klassendiagramm-Trainer im Vollbild öffnen

Merksatz für den Trainer

Einfache Linie = Assoziation
Hohles Dreieck = Vererbung / Generalisierung
Leere Raute = Aggregation
Gefüllte Raute = Komposition
Raute steht am Ganzen
Vererbungs-dreieck zeigt zur Oberklasse
- bedeutet private
+ bedeutet public

Beispiel: Assoziation

Aufgabe:

Ein Kunde kann mehrere Bestellungen haben.
Eine Bestellung gehört zu genau einem Kunden.

UML-Bedeutung:

Kunde ist mit Bestellung verbunden.

Darstellung:

Kunde — Bestellung

Wichtig:

Eine Assoziation ist eine normale Beziehung zwischen Klassen.
Sie wird als einfache Linie dargestellt.

Beispiel: Vererbung

Aufgabe:

Ein Monitor ist ein Produkt.

UML-Bedeutung:

Monitor erbt von Produkt.

Darstellung:

Monitor —|> Produkt

Wichtig:

Das hohle Dreieck zeigt zur Oberklasse Produkt.

Beispiel: Aggregation

Aufgabe:

Ein Team hat mehrere Mitarbeiter.

Ein Mitarbeiter kann aber auch unabhängig vom Team existieren.

UML-Bedeutung:

Team aggregiert Mitarbeiter.

Darstellung:

Team ◇ — Mitarbeiter

Wichtig:

Die leere Raute steht am Ganzen, also bei Team.

Aggregation ist eine schwächere Teil-Ganzes-Beziehung.

Der Teil kann unabhängig vom Ganzen existieren.

Beispiel: Komposition

Aufgabe:

Eine Rechnung besteht aus Rechnungspositionen.
Eine Rechnungsposition gehört fest zu genau einer Rechnung.

UML-Bedeutung:

Rechnung besteht aus Rechnungspositionen.

Darstellung:

Rechnung  — Rechnungsposition

Wichtig:

Die gefüllte Raute steht am Ganzen, also bei Rechnung.
Komposition ist eine starke Teil-Ganzes-Beziehung.
Der Teil ist fest vom Ganzen abhängig.

Wichtiger Unterschied: Aggregation vs. Komposition

Beziehung	Symbol	Bedeutung	Beispiel
Aggregation	leere Raute	schwache Teil-Ganzes-Beziehung	Team hat Mitarbeiter
Komposition	gefüllte Raute	starke Teil-Ganzes-Beziehung	Rechnung besteht aus Positionen

Merksatz:

Leere Raute = Teil kann unabhängig existieren.
Gefüllte Raute = Teil gehört fest zum Ganzen.

Wichtiger Unterschied: Assoziation vs. Vererbung

Beziehung	Symbol	Bedeutung
-----------	--------	-----------

Assoziation	einfache Linie	Klassen stehen miteinander in Beziehung
Vererbung	hohles Dreieck	Unterklasse erbt von Oberklasse

Beispiel Assoziation:

Kunde — Bestellung

Bedeutung:

Kunde und Bestellung hängen fachlich zusammen.

Beispiel Vererbung:

Monitor —▷ Produkt

Bedeutung:

Monitor ist ein spezielles Produkt.

Sichtbarkeit im Trainer

Im Trainer musst du Sichtbarkeiten selbst eintragen.

Zeichen	Bedeutung	Typisch für
+	public / öffentlich	Methoden
-	private / privat	Attribute
#	protected / geschützt	Vererbung / Unterklassen

Beispiel:

- marke: String
+ toString(): String

Wichtig:

Das + oder - soll nicht schon im Feld stehen.
Du sollst es selbst erkennen und eintragen.

Typische Aufgaben im Trainer

Der Trainer enthält Aufgaben zu:

Vererbung:

Monitor ist ein Produkt.

Assoziation:

Kunde hat Bestellungen.

Aggregation:

Team hat Mitarbeiter.

Komposition:

Rechnung besteht aus Rechnungspositionen.

Typische Fehler

Fehler	Warum falsch?
Raute auf der falschen Seite	Die Raute steht immer am Ganzen
Vererbungsdreieck zeigt zur Unterklasse	Das Dreieck muss zur Oberklasse zeigen
Aggregation und Komposition verwechseln	Leere Raute und gefüllte Raute haben unterschiedliche Bedeutung
+ und - vergessen	Sichtbarkeit ist Teil der UML-Notation
Attribute und Methoden vermischen	Attribute speichern Daten, Methoden beschreiben Verhalten
Oberklasse und Unterklasse vertauschen	Gemeinsame Eigenschaften gehören in die Oberklasse

Mini-Testfragen

1. Welche Linie zeigt eine normale Assoziation?

Eine normale Assoziation wird durch eine einfache durchgezogene Linie dargestellt.

Beispiel:

Kunde — Bestellung

2. Welche Pfeilspitze zeigt Vererbung?

Vererbung wird mit einem **hohlen Dreieck** dargestellt.

Wichtig:

Das hohle Dreieck zeigt zur Oberklasse.

Beispiel:

Monitor —▷ Produkt

3. Was bedeutet eine leere Raute?

Eine leere Raute bedeutet **Aggregation**.

Das ist eine schwächere Teil-Ganzes-Beziehung.

Beispiel:

Team ◇— Mitarbeiter

Der Mitarbeiter kann unabhängig vom Team existieren.

4. Was bedeutet eine gefüllte Raute?

Eine gefüllte Raute bedeutet **Komposition**.

Das ist eine starke Teil-Ganzes-Beziehung.

Beispiel:

Rechnung ◆—— Rechnungsposition

Die Rechnungsposition gehört fest zur Rechnung.

5. Wo steht die Raute bei Aggregation oder Komposition?

Die Raute steht am **Ganzen**.

Beispiele:

Team ◆—— Mitarbeiter

Rechnung ◆—— Rechnungsposition

6. Warum soll `+` oder `-` nicht schon im Eingabefeld stehen?

Weil du selbst erkennen und eintragen sollst, ob ein Attribut oder eine Methode öffentlich oder privat ist.

Typisch:

- attribut: Typ

+ methode(): Typ

7. Was bedeutet `- marke: String`?

Das bedeutet:

private Attribut marke vom Typ String

In Java wäre das zum Beispiel:

```
private String marke;
```

8. Was bedeutet `+ toString(): String`?

Das bedeutet:

```
public Methode toString mit Rückgabewert String
```

In Java wäre das zum Beispiel:

```
public String toString()
```

9. Was ist der wichtigste Unterschied zwischen Aggregation und Komposition?

Aggregation:

Teil kann unabhängig existieren.

Komposition:

Teil gehört fest zum Ganzen.

10. Wohin zeigt der Vererbungs Pfeil?

Der Vererbungs Pfeil zeigt immer zur **Oberklasse**.

Beispiel:

```
Monitor —> Produkt
```

Produkt ist die Oberklasse.

Nächste Seite

Danach geht es weiter mit:

```
UML-Aktivitätsdiagramm
```
