Seite 7. UML-Klassendiagramm

UML-Klassendiagramm

Diese Seite erklärt das **UML-Klassendiagramm** so, dass du es schnell für IHK-Aufgaben und Java/OOP-Aufgaben anwenden kannst.

Ein UML-Klassendiagramm wird benutzt, um die **Struktur eines Programms** darzustellen.

Es zeigt:

Klassen
Attribute
Methoden
Sichtbarkeit
Beziehungen zwischen Klassen
Vererbung

Grafik 1: UML-Klassendiagramm – Grundlagen

Diese Grafik zeigt dir die wichtigsten Bausteine eines UML-Klassendiagramms:

- Klassenname
- Attribute
- Methoden
- Sichtbarkeit mit `+`, `-` und `#`
- Assoziation
- Vererbung

Wofür braucht man ein UML-Klassendiagramm?

Ein UML-Klassendiagramm hilft dir, die Struktur eines objektorientierten Programms zu planen.

Beispiel:

Es soll eine Produktverwaltung erstellt werden.
Ein Produkt hat eine Marke, ein Modell und einen Preis.
Monitore und Tastaturen sind spezielle Produkte.

Daraus erkennt man:

- Es gibt Klassen.
- Klassen haben Attribute.
- Klassen haben Methoden.
- Manche Klassen hängen miteinander zusammen.
- Manche Klassen können von anderen Klassen erben.

Woran erkenne ich in einer Aufgabe, dass ein UML-Klassendiagramm gemeint ist?

Typische Signalwörter:

Signalwort / Formulierung	Hinweis
Klasse	Es geht um OOP / Programmstruktur
Attribut	Eigenschaft einer Klasse
Methode	Funktion / Verhalten einer Klasse
Objekt	konkrete Instanz einer Klasse
Vererbung	Oberklasse / Unterklasse
Assoziation	Beziehung zwischen Klassen
Java-Klassen	meist UML-Klassendiagramm
„modellieren Sie die Klassenstruktur“	Klassendiagramm
„stellen Sie Attribute und Methoden dar“	Klassendiagramm

Aufbau einer Klasse

Eine Klasse wird im UML-Klassendiagramm meistens als Rechteck mit drei Bereichen dargestellt:

Beispiel:

Klassenname

Der Klassenname steht oben im Klassendiagramm.

Beispiele:

```
Produkt
Monitor
Tastatur
Kunde
Bestellung
Benutzer
```

Merksatz:

Klassennamen werden meistens großgeschrieben und stehen im Singular.

Also eher:

```
Produkt
```

nicht:

```
Produkte
```

Attribute

Attribute beschreiben die Eigenschaften einer Klasse.

Beispiel Klasse **Produkt**:

```
- marke: String
- modell: String
- preis: double
```

Das bedeutet:

Attribut	Datentyp	Bedeutung
marke	String	Marke des Produkts
modell	String	Modellbezeichnung
preis	double	Preis

Merksatz:

Attribute sind das, was ein Objekt speichert.

Methoden

Methoden beschreiben, was ein Objekt tun kann.

Beispiel:

```
+ getPreis(): double
+ setPreis(preis: double): void
+ toString(): String
```

Aufbau einer Methode:

```
sichtbarkeit methodenname(parameter): rückgabetyp
```

Beispiel:

```
+ setPreis(preis: double): void
```

Bedeutung:

Teil	Bedeutung
+	public

Teil	Bedeutung
`setPreis`	Methodenname
`preis: double`	Parameter
`void`	kein Rückgabewert

Sichtbarkeit

Die Sichtbarkeit zeigt, von wo aus auf Attribute oder Methoden zugegriffen werden darf.

Zeichen	Bedeutung	Erklärung
`+`	public	öffentlich zugreifbar
`-`	private	nur innerhalb der Klasse
`#`	protected	Klasse und Unterklassen
`~`	package	innerhalb des Pakets

Für IHK und Java-Grundlagen sind besonders wichtig:

```
+ public
- private
# protected
```

Typisch in Java:

```
private String marke;
public String toString()
```

UML:

```
- marke: String
+ toString(): String
```

Attribute sind meistens private

In Java sind Attribute meistens private.

Beispiel Java:

```
private String marke;
private String modell;
private double preis;
```

UML:

```
- marke: String
- modell: String
- preis: double
```

Warum?

Damit nicht jeder direkt von außen auf die Daten zugreifen kann.

Das nennt man **Kapselung**.

Methoden sind oft public

Methoden, die von außen genutzt werden sollen, sind meistens public.

Beispiel Java:

```
public String toString() {
    return marke + " " + modell;
}
```

UML:

```
+ toString(): String
```

Grafik 2: Beispiel Produktverwaltung

UML-Klassendiagramm – Beispiel Produktverwaltung

Diese Grafik zeigt ein typisches UML-Klassendiagramm für eine Produktverwaltung:

- **Produkt** ist die Oberklasse
- **Monitor** ist eine spezielle Produktart
- **Tastatur** ist eine spezielle Produktart
- gemeinsame Attribute gehören in die Oberklasse

- spezielle Attribute gehören in die Unterklassen

Beispiel: Produkt als Oberklasse

Wenn mehrere Produktarten gemeinsame Eigenschaften haben, kann man eine Oberklasse verwenden.

Beispiel:

```
Produkt
- marke
- modell
- preis
```

Diese Attribute können für viele Produktarten gelten:

```
Monitor
Tastatur
Maus
CPU
```

Deshalb gehören sie in die gemeinsame Oberklasse **Produkt**.

Vererbung

Vererbung bedeutet:

```
Eine Unterklasse übernimmt Eigenschaften und Methoden einer Oberklasse.
```

Beispiel:

```
Monitor erbt von Produkt.
Tastatur erbt von Produkt.
```

Das bedeutet:

```
Monitor ist ein Produkt.
Tastatur ist ein Produkt.
```

UML-Darstellung:

```
Monitor ———▷ Produkt
Tastatur ———▷ Produkt
```

Der Pfeil zeigt zur **Oberklasse**.

Merksatz:

Der Vererbungs-Pfeil zeigt immer zur allgemeineren Klasse.

Oberklasse und Unterklasse

Begriff	Bedeutung	Beispiel
Oberklasse	allgemeine Klasse	Produkt
Unterklasse	spezielle Klasse	Monitor
Vererbung	Unterklasse übernimmt von Oberklasse	Monitor erbt von Produkt

Beispiel:

```
Produkt
- marke
- modell
- preis

Monitor
- groesseZoll
- aufloesung
```

Der Monitor hat dann fachlich:

```
marke
modell
preis
groesseZoll
aufloesung
```

Assoziation

Eine Assoziation ist eine normale Beziehung zwischen Klassen.

Beispiel:

```
Kunde gibt Bestellung auf
```

Als Klassendiagramm:

```
Kunde ——————— Bestellung
```

Eine Assoziation bedeutet:

```
Diese Klassen stehen miteinander in Beziehung.
```

Multiplizität

Die Multiplizität zeigt, wie viele Objekte miteinander verbunden sein können.

Multiplizität	Bedeutung
`1`	genau eins
`0..1`	kein oder eins
`*`	beliebig viele
`1..*`	mindestens eins
`0..*`	beliebig viele oder keine

Beispiel:

```
Kunde 1 ——— 0..* Bestellung
```

Bedeutung:

```
Ein Kunde kann keine, eine oder mehrere Bestellungen haben.
Eine Bestellung gehört zu genau einem Kunden.
```

Unterschied Kardinalität und Multiplizität

In ERM sagt man oft:

```
1:n
n:m
```

Im UML-Klassendiagramm sieht man eher:

```
1
0..*
1..*
*
```

Beispiel:

```
Kunde 1 ——— 0..* Bestellung
```

entspricht ungefähr:

```
Kunde 1:n Bestellung
```

Aggregation und Komposition kurz erklärt

Diese beiden Beziehungen können in UML vorkommen, sind aber für den schnellen IHK-Einstieg nicht immer der Schwerpunkt.

Beziehung	Symbol	Bedeutung
Aggregation	leere Raute	Teil-Ganzes-Beziehung, Teil kann auch ohne Ganzes existieren
Komposition	gefüllte Raute	starke Teil-Ganzes-Beziehung, Teil gehört fest zum Ganzen

Einfacher Merksatz:

```
Aggregation = hat-Beziehung, Teil kann unabhängig existieren
Komposition = besteht-aus-Beziehung, Teil ist stark abhängig
```

Beispiel Aggregation:

```
Team hat Mitarbeiter
```

Ein Mitarbeiter kann auch ohne dieses Team existieren.

Beispiel Komposition:

```
Rechnung besteht aus Rechnungspositionen
```

Eine Rechnungsposition ergibt ohne Rechnung meist keinen Sinn.

Unterschied Klasse und Objekt

Begriff	Bedeutung	Beispiel
Klasse	Bauplan	Produkt
Objekt	konkrete Ausprägung	Produkt p1 = neuer Monitor

Beispiel:

```
Klasse: Produkt
Objekt: monitor1
```

Java:

```
Produkt monitor1 = new Produkt("Dell", "U2723QE", 499.99);
```

Die Klasse beschreibt, welche Attribute und Methoden ein Objekt haben kann.

Das Objekt ist eine konkrete Instanz dieser Klasse.

UML-Klassendiagramm vs. ERM

UML-Klassendiagramm	ERM
zeigt Programmstruktur	zeigt Datenmodell
Klassen	Entitäten
Attribute	Attribute
Methoden	keine Methoden
Vererbung möglich	normalerweise keine Methoden/Vererbung
wichtig für OOP / Java	wichtig für Datenbanken

Merksatz:

```
ERM = Daten fachlich modellieren
UML-Klassendiagramm = Programmklassen modellieren
```

Vorgehensweise in der Prüfung

Wenn du ein UML-Klassendiagramm erstellen sollst, gehe so vor:

Schritt	Frage
1	Welche Klassen gibt es?
2	Welche Attribute gehören zu welcher Klasse?
3	Welche Methoden sind sinnvoll oder vorgegeben?
4	Gibt es gemeinsame Eigenschaften?
5	Kann man eine Oberklasse bilden?
6	Gibt es Vererbung?
7	Gibt es Beziehungen zwischen Klassen?
8	Welche Multiplizitäten sind sinnvoll?
9	Welche Sichtbarkeit brauchen Attribute und Methoden?

Beispiel: Produktverwaltung

Fachliche Beschreibung:

```
Es soll eine Produktverwaltung erstellt werden.
Jedes Produkt hat eine Marke, ein Modell und einen Preis.
Ein Monitor hat zusätzlich eine Größe in Zoll und eine Auflösung.
Eine Tastatur hat zusätzlich ein Layout und die Information, ob sie mechanisch ist.
```

Mögliche Klassen:

```
Produkt
Monitor
Tastatur
```

Oberklasse:

```
Produkt
```

Unterklassen:

```
Monitor
Tastatur
```

Warum?

```
Monitor ist ein Produkt.
Tastatur ist ein Produkt.
```

Mögliches Klassendiagramm als Text

```
Produkt
--------------------------------
- marke: String
- modell: String
- preis: double
--------------------------------
+ toString(): String


Monitor extends Produkt
--------------------------------
- groesseZoll: double
- aufloesung: String
--------------------------------
+ toString(): String


Tastatur extends Produkt
--------------------------------
- layout: String
- mechanisch: boolean
--------------------------------
+ toString(): String
```

Typische IHK-Fehler beim UML-Klassendiagramm

Fehler	Warum problematisch?
Attribute und Methoden vermischen	Attribut speichert Daten, Methode führt Verhalten aus
Sichtbarkeit vergessen	`+`, `-`, `#` sind wichtige UML-Informationen
Attribute öffentlich machen	In Java sind Attribute meist private
Vererbung falsch herum zeichnen	Pfeil zeigt zur Oberklasse
Klassen im Plural schreiben	Klassennamen meist Singular
ERM und Klassendiagramm verwechseln	ERM hat keine Methoden
Methoden ohne Rückgabetyp schreiben	Rückgabetyp gehört häufig dazu
Unterklasse enthält alle Oberklassenattribute doppelt	Gemeinsames gehört in die Oberklasse

---

Prüfungs-Merksätze

```
Klasse = Bauplan für Objekte
Objekt = konkrete Instanz einer Klasse
Attribut = gespeicherte Eigenschaft
Methode = Verhalten / Funktion
+ = public
- = private
# = protected
Vererbungspfeil zeigt zur Oberklasse
Gemeinsame Attribute gehören in die Oberklasse
Spezielle Attribute gehören in die Unterklasse
ERM zeigt Daten, Klassendiagramm zeigt Programmstruktur
```

Mini-Beispiel 1

Aufgabe:

```
Ein Produkt hat eine Marke, ein Modell und einen Preis.
```

Lösung:

```
Produkt
--------------------------------
- marke: String
- modell: String
```

```
- preis: double
```

Begründung:

```
marke, modell und preis sind Eigenschaften eines Produkts.
Deshalb sind sie Attribute.
```

Mini-Beispiel 2

Aufgabe:

```
Ein Monitor ist ein Produkt und hat zusätzlich eine Größe in Zoll.
```

Lösung:

```
Monitor erbt von Produkt.

Monitor
-------------------------------
- groesseZoll: double
```

Begründung:

```
Monitor ist eine spezielle Produktart.
Deshalb ist Vererbung sinnvoll.
```

Mini-Beispiel 3

Aufgabe:

```
Ein Kunde kann mehrere Bestellungen haben.
```

Mögliches UML-Klassendiagramm:

```
Kunde 1 ——— 0..* Bestellung
```

Begründung:

Ein Kunde kann mehrere Bestellungen haben.

Eine Bestellung gehört zu einem Kunden.

Mini-Testfragen

1. Wofür wird ein UML-Klassendiagramm verwendet?

Ein UML-Klassendiagramm wird verwendet, um die Struktur eines objektorientierten Programms darzustellen.

Es zeigt:

Klassen
Attribute
Methoden
Beziehungen
Vererbung

2. Was ist eine Klasse?

Eine Klasse ist ein Bauplan für Objekte.

Beispiel:

Produkt
Monitor
Tastatur

3. Was ist ein Attribut?

Ein Attribut ist eine gespeicherte Eigenschaft einer Klasse.

Beispiel:

- marke: String
- preis: double

4. Was ist eine Methode?

Eine Methode beschreibt ein Verhalten oder eine Funktion einer Klasse.

Beispiel:

```
+ toString(): String
+ getPreis(): double
```

5. Was bedeutet `+` im UML-Klassendiagramm?

`+` bedeutet:

```
public / öffentlich
```

Das Element ist von außen zugreifbar.

6. Was bedeutet `-` im UML-Klassendiagramm?

`-` bedeutet:

```
private / privat
```

Das Element ist nur innerhalb der Klasse direkt zugreifbar.

7. In welche Richtung zeigt der Vererbungspfeil?

Der Vererbungspfeil zeigt zur **Oberklasse**.

Beispiel:

```
Monitor ──▷ Produkt
```

8. Warum gehören marke, modell und preis in die Klasse Produkt?

Weil diese Attribute für mehrere Produktarten gemeinsam gelten können.

Beispiele:

```
Monitor
Tastatur
Maus
CPU
```

Alle können Marke, Modell und Preis haben.

9. Was ist der Unterschied zwischen Klasse und Objekt?

Eine Klasse ist der Bauplan.

Ein Objekt ist eine konkrete Instanz dieser Klasse.

Beispiel:

```
Klasse: Produkt
Objekt: monitor1
```

10. Was ist der Unterschied zwischen ERM und UML-Klassendiagramm?

ERM modelliert Daten fachlich.

UML-Klassendiagramm modelliert die Programmstruktur.

Wichtig:

```
Ein Klassendiagramm kann Methoden und Vererbung enthalten.
Ein ERM normalerweise nicht.
```

Nächste Seite

Danach kommt die eigene Trainer-Seite:

UML-Klassendiagramm-Trainer

Dort bauen wir den interaktiven Trainer mit Aufgaben zu:

- Klassen erkennen
- Attribute zuordnen
- Methoden eintragen
- Sichtbarkeit bestimmen
- Vererbung erkennen
- Beziehungen zwischen Klassen darstellen

Revision #2
Created 27 May 2026 23:13:52 by Admin
Updated 28 May 2026 09:25:50 by Admin