V6 Autoklass

In dieser Lektion sprechen wir über **Kompositions-Beziehungen**

Komposition ist eine strengere Form der Aggregation, bei der eine Klasse Objekte einer anderen Klasse enthält, die jedoch ohne die Containerklasse nicht existieren können. Die Lebensdauer der enthaltenen Objekte ist vollständig an die Lebensdauer der Containerklasse gebunden.

Ein Beispiel dafür ist die Beziehung zwischen **Auto** und **Reifen**. Wenn das Auto zerstört oder verbrannt wird, werden auch die Reifen damit vernichtet.

```
void main() {
 final maxFuehrerschein = Fuehrerschein(
    id: 123456,
    typ: FuehrerscheinTyp.pkw,
    ausstellungsdatum: DateTime(2020),
    ablaufdatum: DateTime(2025),
  );
  final max = Fahrer(vorname: 'Max', nachname: 'Mustermann', alter: 25,
fuehrerschein: maxFuehrerschein);
  final auto = AutoV6(baujahr: DateTime(2020), marke: 'Mercedes', reifenRadius:
28, reifenBreite: 18, fahrer: max);
}
class AutoV6 {
  static const String material = 'Metal';
  int _reifenZahl = 4;
  int _maxInsasseZahl;
  final int _minIinsasseZahl = 1;
  final DateTime _baujahr;
  double _reifenRadius;
  double _reifenBreite;
  Fahrer fahrer;
  String? marke;
  Reifen reifen;
  // named Konstruktor
  AutoV6({
    required DateTime baujahr,
    required this.marke,
    int maxIinsasseZahl = 5,
    required double reifenRadius,
    required double reifenBreite,
    required this.fahrer,
  })  : _maxInsasseZahl = maxIinsasseZahl,
        _baujahr = baujahr,
        _reifenRadius = reifenRadius,
        _reifenBreite = reifenBreite,
        reifen = Reifen(radius: reifenRadius, breite: reifenBreite, position: '');
        }
```

Wie Sie sehen, werden **Reifen** nicht im **Main**-Bereich erstellt und zum Konstruktor eingefügt, wie es bei **Fahrer** und **Führerschein** der Fall ist. Stattdessen werden sie direkt im Konstruktor der **Auto**-Klasse selbst erstellt. Das bedeutet, dass, wenn das **Auto**-Objekt zerstört wird, auch die **Reifen**-Objekte damit vernichtet werden. Dies verdeutlicht die starke Abhängigkeit der **Reifen** von der Lebensdauer des **Auto**-Objekts in der Kompositions-Beziehung.

Multi-Relationen zwischen verschiedenen Klassen

Als weiteres Beispiel für Beziehungen zwischen Klassen kann man die **Fahrt**-Klasse definieren.

Das **Fahrt**-Objekt kann nur erstellt werden, wenn ein **Auto** fährt (also wenn die Methode `fahren()` aufgerufen wird). Außerdem benötigt es ein **Fahrer**-Attribut.

Das bedeutet, dass der Aufbau dieser Klasse eine Methode der **Auto**-Klasse und ein Objekt der **Fahrer**-Klasse erfordert.

Relationen

1. Die Beziehung zwischen **Fahrt** und **Fahrer** ist eine **Aggregation**, da der **Fahrer** auch ohne eine **Fahrt** existieren kann.
2. Die Beziehung zwischen **Auto** und **Fahrt** ist eine **Komposition**, da die **Fahrt** nicht ohne das **Auto** existieren kann.

Diese Beziehungen zeigen, wie Klassen in verschiedenen Formen miteinander verbunden sein können.

Aufgabe

Erstellen Sie eine **Arbeitsstelle**-Klasse. Diese Klasse muss die folgenden Attribute enthalten:

- **Gehalt**
- **Arbeitszeit**
- **Berufsname**

Die **Mensch**-Klasse soll ein **Arbeitsstelle**-Objekt beinhalten, wobei die Beziehung zwischen **Mensch** und **Arbeitsstelle** durch eine **Kompositionsrelation** definiert wird. Dies macht Sinn, da, wenn der Mensch stirbt, auch seine Arbeitsstelle endet.

Hinweis

Standardmäßig hat der Mensch keine Arbeitsstelle, aber durch die Methode ArbeitsstelleFinden() erhält man eine.