**Objektorientierung**

Klassen - bilden Blueprint für Objekterzeugung

- Nach einem Blueprint können beliebig viele Objekte erzeugt werden

Objekte - stellen die sog. Instanz einer Klasse da

- können erzeugt, verändert und zerstört werden

- haben eine Identität und einen Zustand

Attribute - beschreiben die Eigenschaften von Objekten

- beschreiben Zustand des Objekts

Methoden - beschreiben das Verhalten von Objekten

- stellen Funktionalität von Objekten da

**Attribute und Methoden**

Attribute

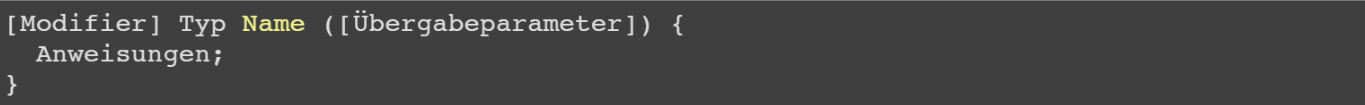
- sind Konstanten und Variablen innerhalb eines Objekts und werden wie solche deklariert

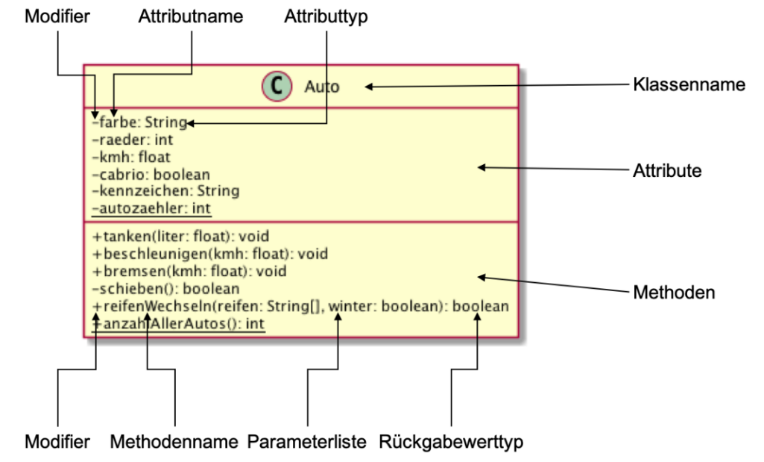
Methoden

- sind die Algorithmen innerhalb eines Objekts

- können einen Rückgabewert haben, der einfach oder Referenzdatentyp sein kann oder void

- void Methoden haben keinen Rückgabewert



**Klassendiagramm in UML**

**Objekte erzeugen**

Objekte sind Instanzen einer Klasse, d.h. man muss um sie zu erzeugen zunächst eine Klasse definieren

Definition einer Klasse

- beginnt mit „class“

- innerhalb der Klasse werden Attribute und Methoden implementiert

- beschreibt Aufbau und Verhalten einer Klasse

- Variable vom Typ der Klasse wird deklariert

- mit „new“ wird ein neues Objekt der Klasse instanziiert und musss der Variable zugewiesen werden

- „new“ allokier Speicher und ruf implizit Konstruktor auf

Konstruktoren

- spezielle Methode zur Erzeugen von Objekten und tragen den Klassenamen

- werden ausschließlich von „new“ aufgerufen

- liefern als Ergebnis Referenz auf neues Objekt

- Standard Konstruktor (ohne Übergabeparameter) wird aufgerufen wenn kein anderer implementiert wird

- kann benutzt werden um Attribute zu initialisieren

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

**Kapselung**

- Grundsatz der objektorientierten Programmierung keine Eigenschaft Javas

- Idee: auf Attribute wird nicht direkt, sondern über spezielle Methoden zugegriffen

- die speziellen Methoden heißen getter- und setter-Methoden

- dazu muss man die Attribute mit Sichbarkeitmodifiern verbergen

- Ziel: Attribute sollen nur im Sinne des Entwicklers verändert werden →Plausibilitätsprüfung

Sichtbarkeitsmodifier

Private - nur innerhalb der Klasse sichtbar

- stärkste Form der Kapslung, da nur in Klasse nutzbar

Protected - nur innerhalb eines Paketes und Subklassen sichtbar

Default - nur in Paket sichtbar

Public - von überall sichtbar

- normalerweise sind getter und setter public um auf private Attribute zuzugreifen

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Zugriff auf Attribute und Methoden

- per Punktnotation (objektname.methode(übergabeparameter);)

- geht nur wenn die Sichtbarkeit es zulässt

Call by value - Parameter warden in Java mit Call by Value übergeben

- Änderung der Parameter in der Methode hat keinen Einfluss auf Übergabeparameter

Call by reference - werden Referenzdatentypen übergeben steht das Orginalobjekt inklusive Attribute und Methoden in der Methode zur Verfügung

- Änderung der Attribute finden im Orginalobjekt statt

Überladene Methoden

- Methoden mit demselben Namenwerden innerhalb einer Klasse mehrmals definiert

- unterscheiden sich in Anzahl und/oder Typisierung der Übergabeparameter

- Modifier können die Sichtbarkeit verändern

- Ziel: Gleichnamige Methoden können mit unterschiedlichen Übergabeparametern versorgt werden und so unterschiedliche Funktionalitäten erfüllen

Klassenattribute und -Methoden

- existieren unabhängig von einer Instanz einer Klasse

- können verwendet werden ohne, dass ein Objekt existiert

- werden durch den Modifier „static“ als Klassenattribut bzw. -methode defniert

- Static-Methoden können nur auf Static-Attribute zugreifen

- Zugriff ebenfalls über Punktnotation

- werden in UML unterstrichen dargestellt

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungEnumeration

- Spezielle Klasse (erbt von java.lang.Enum

- definiert eine Menge an gültigen Werten

- bietet Prüfung auf gültige Werte zur Laufzeit

Garbage-Collector

- nicht mehr referenzierte Objekte (nicht mehr gebraucht) werden unreachable

- Java hat einen automatischen Garbage Collector (startet wenn Speicherplatz benötigt wird)

- kann explizit über System.gc() gestartet werden

- Destruktoren sind parameterlose Methoden mit dem namen „finalize“

Assoziation

- beschreibt eine Menge an Beziehungen zwischen Objekten

- binäre Assoziationen beschreiben die Beziehungen zwischen zwei Klassen

- können durch verbale Beschreibung der Beziehung benannt werden

- Darstellung in UML durch einfache Verbindungslinie

Aggregation

- stellt spezielle Art der Assoziation dar (Teil von)

- beschreiben keine existentiellen Abhängigkeiten (Wagon auch ohne Ladung)

- Darstellung in UML mit Verbindungslinie mit Raute bei der Aggregationsklasse (Ganze)

Komposition

- stellt spezielle Art der Aggregation dar (Teil von)

- beschreibt existentielle Abhängigkeiten (Auto nicht ohne Motor und Chassis)

- Darstellung in UML wie Aggregation nur ausgefüllte Raute