

Objektorientierte Programmierung

- Partnerarbeit: Startet die Datei **pong.jar** und spielt ein wenig gegeneinander
 - Spieler oben: Pfeiltasten links/rechts
 - Spieler unten: A und D
- Beantwortet folgende Fragen:
 - Welche “Spielobjekte” gibt es?
 - Welche Eigenschaften und Fähigkeiten dieser Objekte lassen sich beobachten?

- Grundidee:
 - Fasse Daten und die zugehörigen/darauf anwendbaren Methoden zu einem **Objekt** zusammen
- Objekt:
 - Verfügt über **Attribute** (Eigenschaften) und **Methoden** (Fähigkeiten)
- Klasse:
 - “Bauplan” für Objekte eines bestimmten Typs
 - Legt fest, welche Attribute Objekte dieses Typs haben werden, ohne deren konkrete Werte festzulegen
 - Definiert die Methoden, die zu Objekten dieses Typs gehören
- Von einer Klasse lassen sich also in der Regel beliebig viele Objekte erzeugen!
 - Man nennt diese Objekte dann auch **Instanz** der entsprechenden Klasse

UML

- UML (“Unified Modeling Language”)
 - Sprache zur (graphischen) Modellierung von Programmen/Systemen
 - Vielzahl von unterschiedlichen Diagrammarten für verschiedenste Anwendungszwecke
- Für uns zunächst wichtig:
 - Klassendiagramme
 - Objektdiagramme

- Stellen den Aufbau einer bestimmten Klasse dar
 - Klassenname
 - Attribute
 - Methoden
 - Zugriffsrechte (später)
- Vorsicht:
 - Andere Syntax zur Angabe von Datentyp/Rückgabetyt als bei Java!

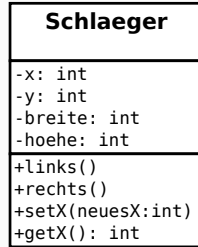


Abbildung 1: Ein Klassendiagramm

- Stellen ein konkretes Objekt dar
 - Objektname und Klassenname
 - Konkrete Werte aller oder mancher Attribute zu einem bestimmten Zeitpunkt
 - Keine Methoden (warum nicht?)

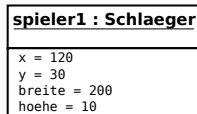


Abbildung 2: Ein Objektdiagramm

Aufgabe 2

- Gruppen aus 2 bis 3 Personen bilden
- Beispielszenario wählen und für das Szenario:
 - geeignete Klasse(n) mit Attributen und Methoden identifizieren
 - Klasse(n) in einem UML-Klassendiagramm darstellen
 - Ein Objektdiagramm zeichnen
 - Ergebnisse kurz im Plenum vorstellen

- Hinweis:

Es geht ausschließlich darum, eine (oder einige wenige) wichtige Klasse(n) zu identifizieren und darzustellen, es soll also keine vollständige Programmarchitektur entwickelt werden. Insbesondere müsst ihr euch keine Gedanken machen, wie die einzelnen Objekte verwaltet werden können.

1. Eine Software zur Verwaltung von Schülern und Schulklassen
2. Ein Geometrieprogramm, das das Arbeiten mit verschiedenen Figuren (Kreis, Rechteck, ...) erlaubt
3. Eine Software, mit der ein Autohändler seine Autos verwaltet
4. Eine Software zur Verwaltung des Warenbestands eines Supermarkts
5. Eine digitale Fernsehzeitschrift
6. Das Spiel "Schach"