Objektorientierung

Grundlagen
und
Modellierung

Inhalt

- Motivation
- Was sind Objekte
- Grundeigenschaften Objektorientierter Programmierung
- Schnellkurs zu UML (Unified Modelling Language)

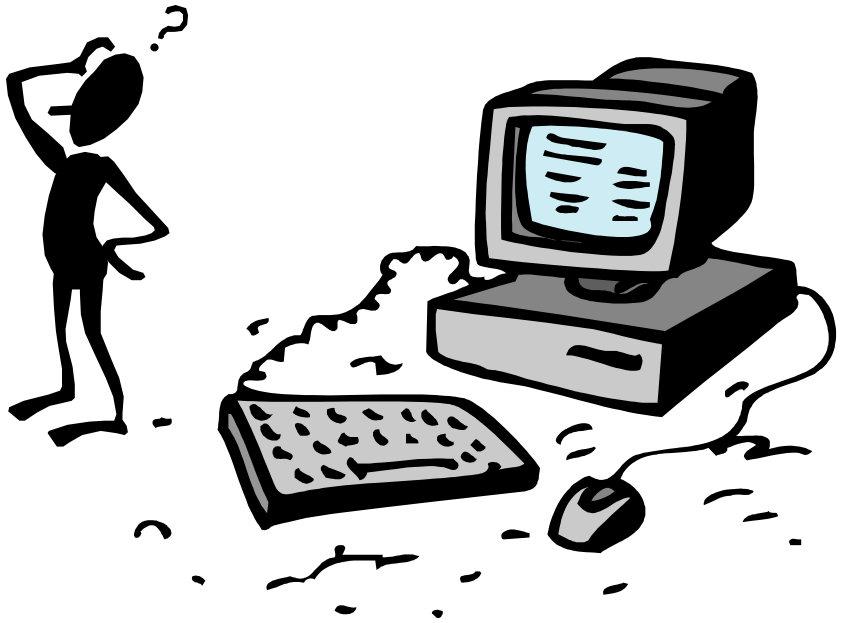
Motivation

- Die Bedeutung der Objektorientierung ist nicht auf ein reines Programmierkonzept beschränkt. Sie stellt vielmehr einen eigenen Denkstil, eine bestimmte Herangehensweise an Problemstellungen dar. (Erler,Ricken)

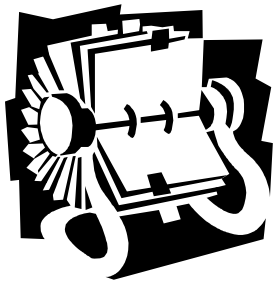
Was ist ein Rechner, genaugenommen ?

- Bits, Bytes, Speicherzellen zur Zahlendarstellung **à** als element. Objekte
- o.g. Objekte kopieren, mittels mathematischer Funktionen verknüpfen **à** gemäß einer Programmvorschrift

- ein Rechner war ursprünglich für die Lösung mathematischer Probleme gemacht



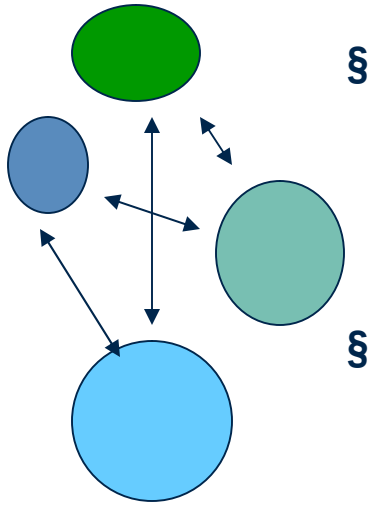
Motivation



Was braucht man eigentlich ?

- ein Gerät zum
  - Schreiben, Malen, Spielen
  - Verwalten von Adressen,…
- ein Buch, in dem man schneller was findet
- natürlich schneller, besser wie bisher
- in diesem Sinne ist ein Rechner die Nachbildung des praktischen Lebens mit anderen Mitteln.

Abstraktion: Modell = Abbild der Wirklichkeit



- Jede Situation aus dem wirklichem Leben lässt sich betrachten als:
- Objekte
  - Arzt, Helferin, Patient
  - Kunde, Konto, Bank
- und Handlungen zwischen diesen Objekten
  - Der Arzt behandelt den Patient
  - Der Kunde eröffnet ein Konto

Neu bei dieser Art der Modellierung (Denkweise) ist,
dass Objekteigenschaften und Handlungen
(d.h. Daten und Methoden) zusammengehören
à das führt auf die folgenden Grundprinzipien

Grundprinzipien der objektorientierten Programmierung

- Kapselung
  - Objekteigenschaften (d.h. Daten) und Methoden bilden eine Einheit. Auf die Daten eines Objektes kann nur über seinen (öffentlichen) Methoden zugegriffen werden.
- Vererbung
  - Ein Objekt kann die Eigenschaften eines anderen Objektes erben. Eigenschaft ist dabei nicht nur Daten (Attribute) sondern auch alle Methoden.
- Polymorphie (Vielgestaltigkeit)
  - Objekte können auf die selbe Nachricht (Methode z.B. drucke()) unterschiedlich reagieren, und das erst zur Laufzeit. D.h. das Interface (die Schnittstelle) bleibt gleich aber die Methode ändert sich bei unterschiedlichen Objekten.

Was ist ein Objekt?

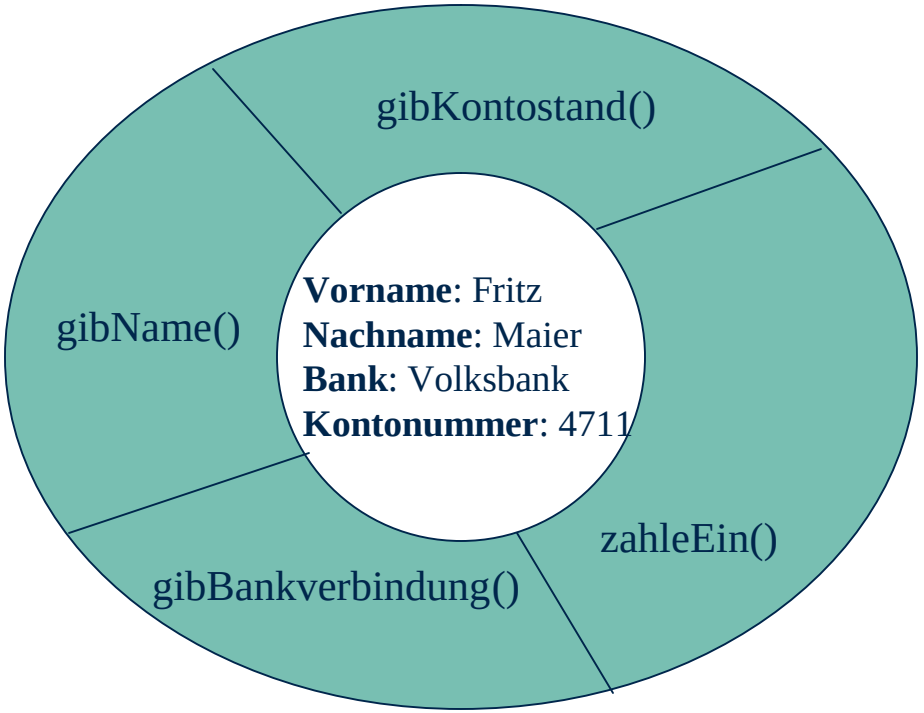
- Ein Objekt ist ein genau definiertes, in sich abgeschlossenes Element.
- Jedes Objekt unterscheidet sich von allen anderen Objekten durch seine Objektidentität (seinen eindeutigen Namen)
- in der realen Welt:
  - der Kunde Hugo Müller,
  - Die Tür des Vorlesungssaals 14/04
- in der EDV ist ein Objekt ein Modell eines Elements eines realen Systems, das durch ein Programm abgebildet werden soll.
  - das Modell des Kunden Müller im EDV-System
  - das Modell einer Tür des Vorlesungssaals

Was ist ein Objekt?

- Ein Objekt besitzt Eigenschaften (**Attribute**) mit bestimmten Ausprägungen (Attributwerten)
- Attribute können zu verschiedenen Zeitpunkten unterschiedliche Werte annehmen (variabler Charakter), sie drücken den Zustand des Objektes aus.
  - Der Vornamen des Kunden Müller
  - Die Farbe der Tür zum Hörsaal
- Ein Objekt besitzt Operationen (**Methoden**), die es durchführen kann
- Operation sind Ausdruck der Dynamik eines Objektes und beschreiben sein Verhalten gegenüber der Umwelt
  - der Kunde Müller kauft ein Produkt
  - die Tür des Hörsaals öffnet sich

Kapselung

- Die Attributwerte eines Objektes sind nur dem Objekt selbst bekannt
- Auf die Attributwerte kann nur über definierte Methode zugegriffen werden. (z.B. zahleEin())
- Wie die Methoden funktionieren (d.h. implementiert sind) ist von Außen nicht erkennbar. Andere Objekte kennen nur die Methode (d.h. das Interface)



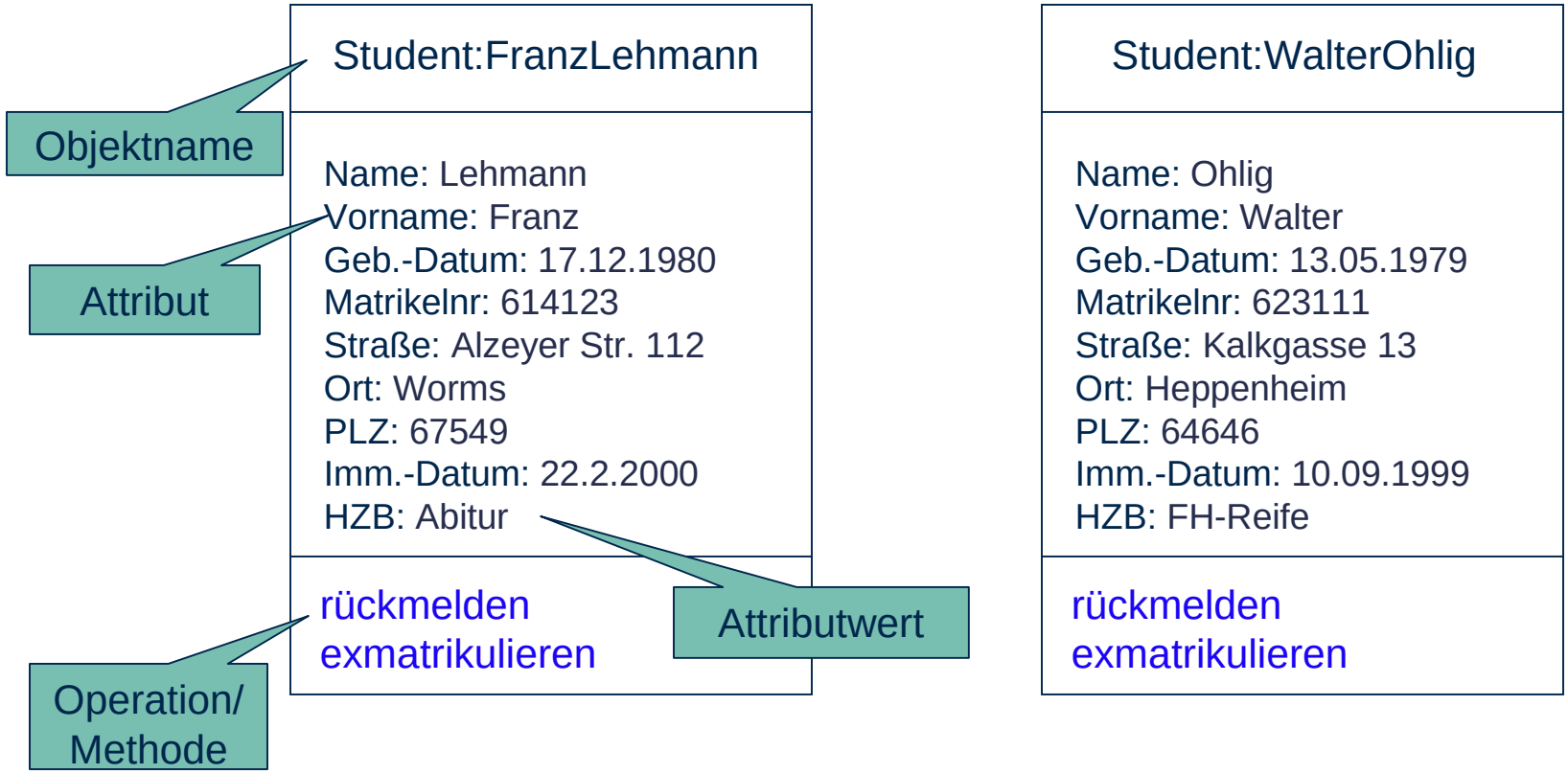
Was ist eine Klasse ?

- ein Bauplan für ein Objekt
- In der Klasse sind alle Attribute und Operationen eines Objekts der Klasse festgelegt
- Objekte der gleichen Klasse unterscheiden sich nur in den Attributwerten
- Ein Objekt ist eine **Instanz** seiner Klasse
  - Kunde
  - Tür
- In der EDV ist eine Klasse ein Bauplan für ein Modell eines Elements eines realen Systems, das durch ein Programm abgebildet werden soll.

Objekt mit UML (Unified Modelling Language)

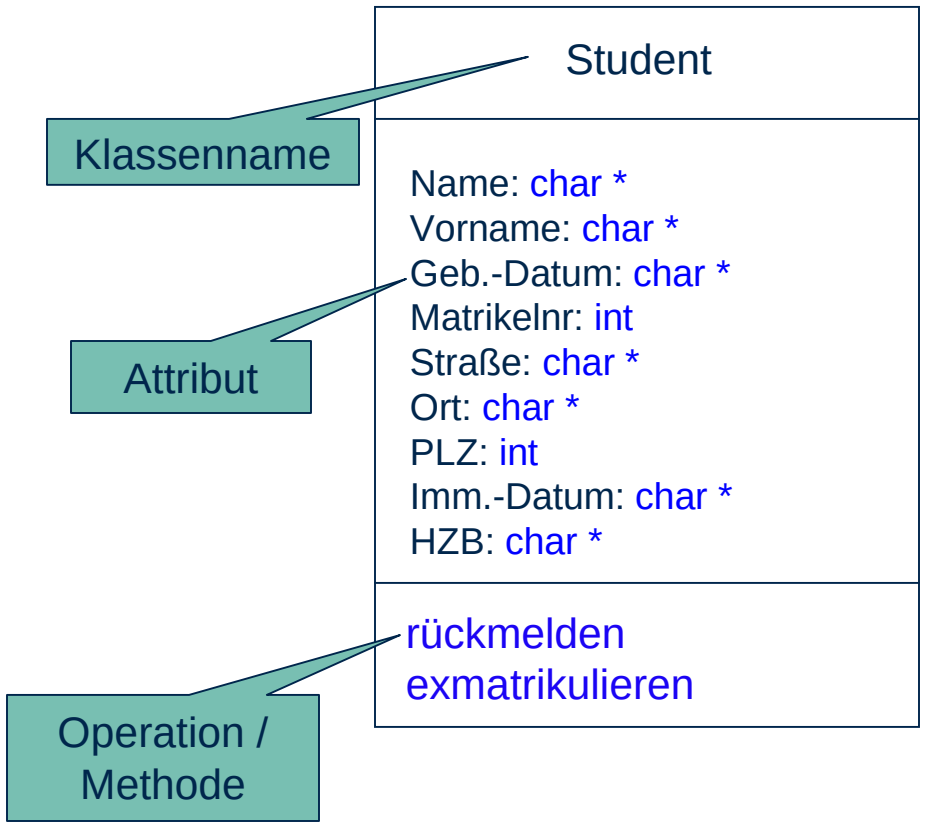
Beispiel für Objekte:

Objektdiagramm



Klasse mit UML

Zusammenfassung der Objekte zur Klasse:



Klassendiagramm

Klasse:
Beschreibung einer Menge von Objekten mit gleichen Attributen, Operationen, Beziehungen und gleicher Bedeutung

Objekt:
Konkrete Ausprägung einer Abstraktion; Instanz einer Klasse

Ereignis:
Zustandsänderung eines oder mehrere Objekte

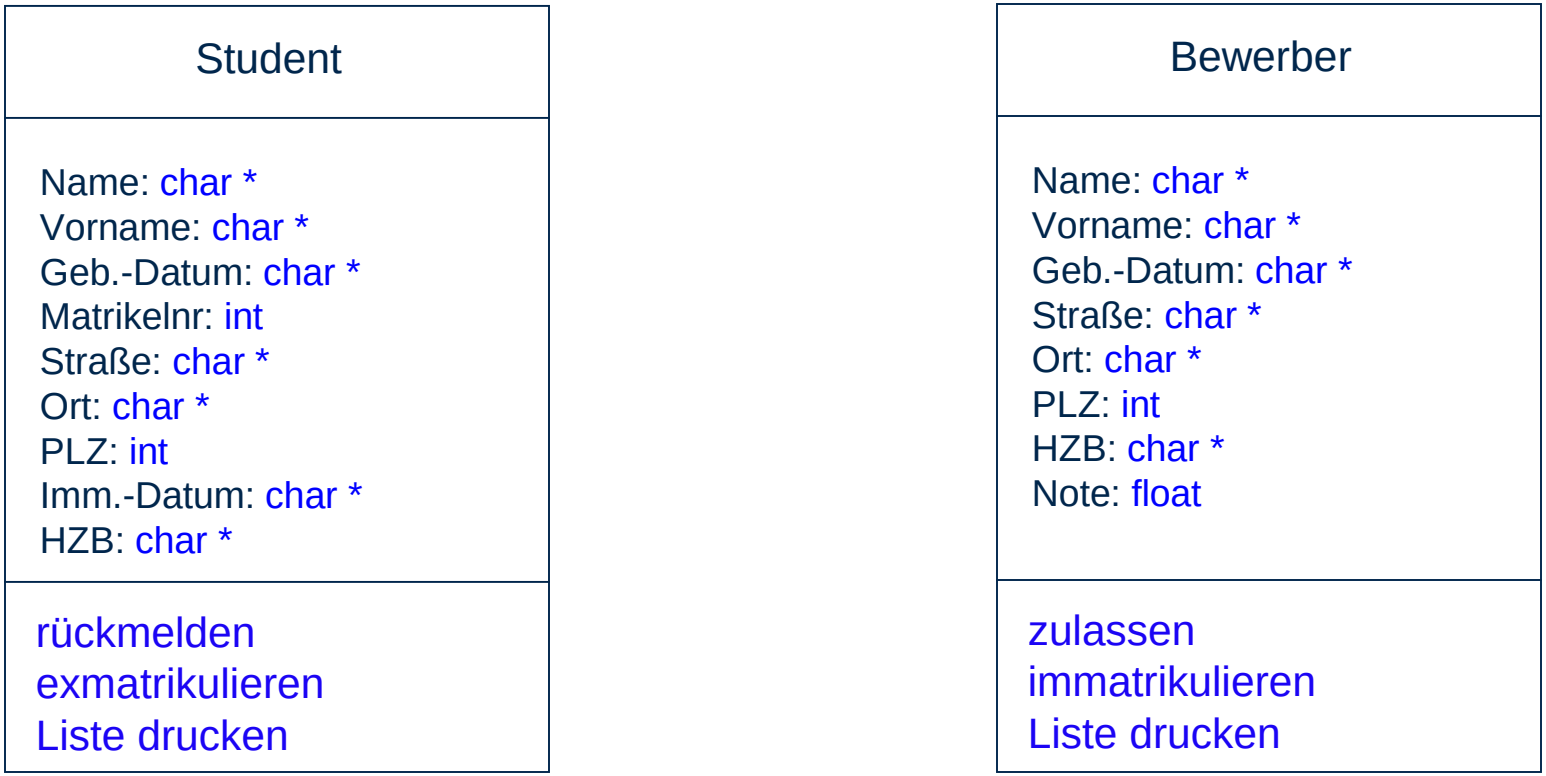
Botschaft:
Aktivierung einer Operation aus einem anderen Prozess

Vererbung

- Klassen können miteinander in "verwandtschaftlichem" Verhältnis stehen
- Eine Klasse (Ober-, Super-, Basis-) kann ihre Attribute und Methoden an andere Klassen (Unter-, Sub-, abgeleitete) "vererben"
- Die Unterklasse beinhaltet genau die gleichen Attribute und Methoden wie die Oberklasse, kann aber noch zusätzliche Attribute und Methoden hinzufügen
- Ein Objekt der Unterklasse ist auch ein Objekt der Oberklasse (d.h. Vererbung ist Spezialisierung)

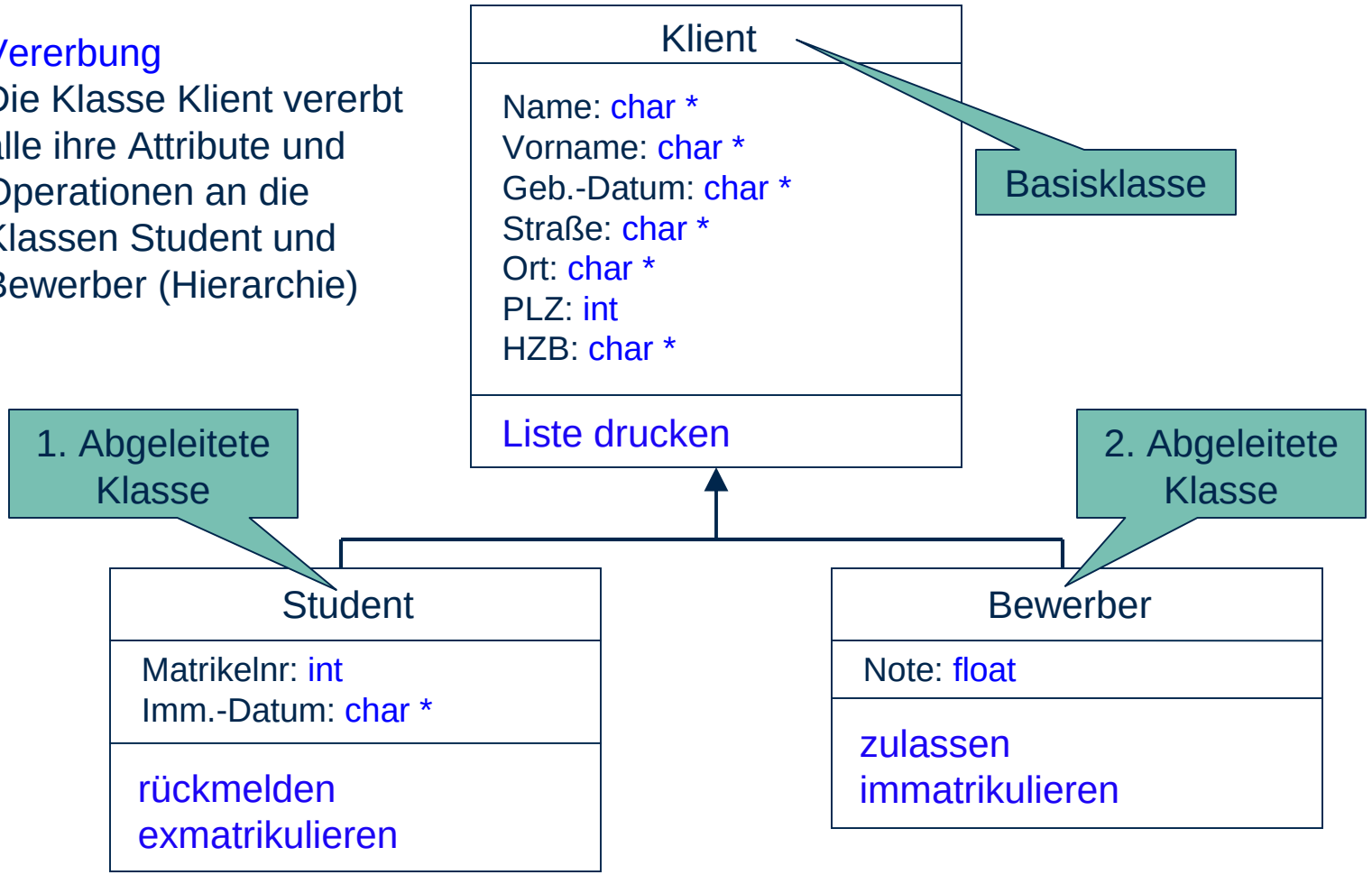
Vererbung (UML)

Zwei Klassen mit gemeinsamen Attributen und Operationen:



Vererbung als Klassendiagramm

Vererbung
Die Klasse Klient vererbt alle ihre Attribute und Operationen an die Klassen Student und Bewerber (Hierarchie)

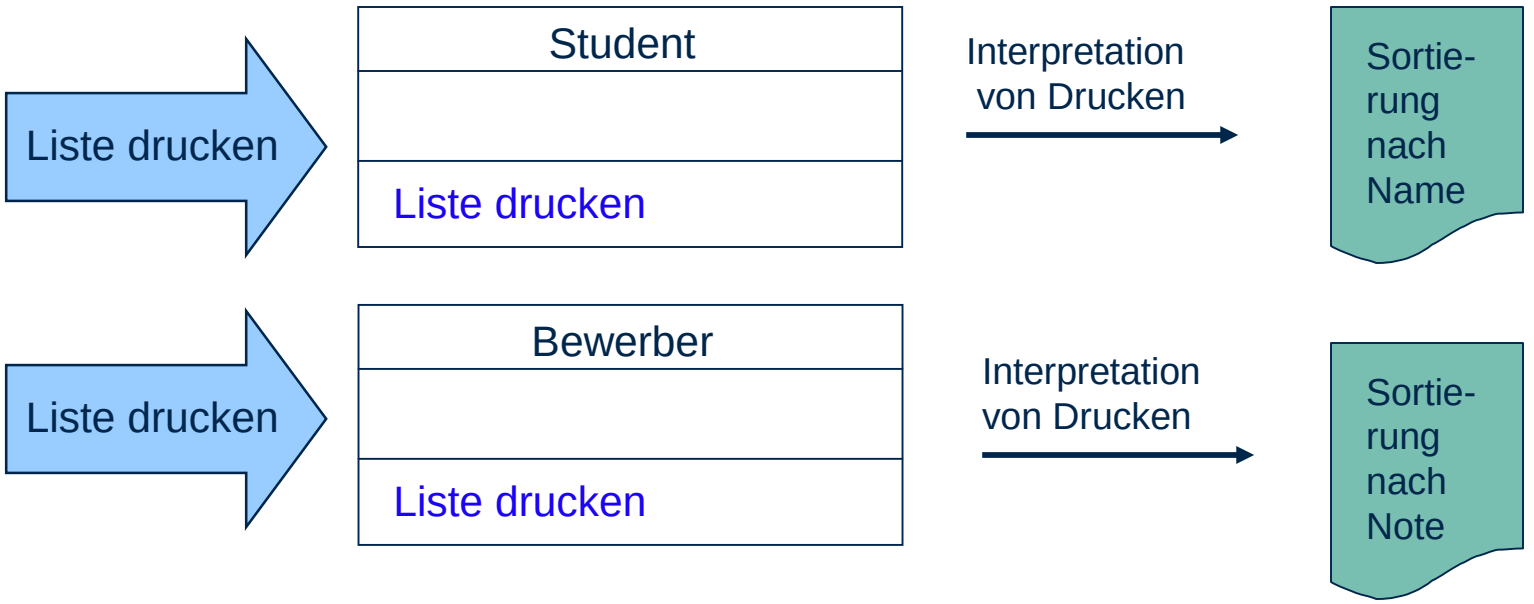


Polymorphie

- Objekte können untereinander agieren indem sie die Methoden des Objektes aufrufen (Nachrichten/Botschaften senden).
  - z.B. ListeDrucken() oder zahleEin()
- Dabei kann die gleiche Botschaft – an Objekte unterschiedlicher Klassen geschickt – zu unterschiedlichen Ergebnissen führen
- Der Auftraggeber muss dabei nicht wissen, welcher Klasse das Empfänger-Objekt angehört
- Dies Bezeichnet man als Vielgestaltigkeit oder Polymorphismus

Ein kleines (nicht programmier) Beispiel ist das Thermostat. Unabhängig vom Heizungssystem (Öl,Gas,…) arbeitet das Thermostat immer gleich. Es ist quasi das Interface – egal welches Heizungssystem verwendet wird

Polymorphy



Polymorphismus:
Die gleiche Botschaft an Objekte verschiedener Klassen einer Vererbungshierarchie kann unterschiedlich interpretiert werden.
Allgemeiner: Scheinbar gleiche Verpackungen verbergen kontextspezifische Inhalte

UML: Begriffe

Dokumentation von Klassen:

Zu jeder Klasse sollen folgende Informationen vorliegen

- Name der Klasse (obligatorisch)
- Namen der Attribute (obligatorisch)
- Typ der Attribute (optional)
- Name der Operationen (obligatorisch)
- Angabe der Klassen, von denen Eigenschaften geerbt werden (obligatorisch)

Standards für textuelle Beschreibungen:

IDL Interface Definition Language der OMG (Object Management Group)

ODL Object Definition Language der ODMG (Object Database Management Group)

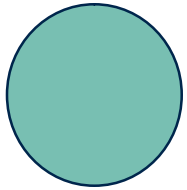
UML: Begriffe

Synonyme:

	OO-SE-Methoden	C++	Microsoft C++
Attribut	Instanzvariable	Datenelement	Member-Variable
Operation	Methode	Elementfunktion	Member-Funktion
Objekt	Instanz	Exemplar	

Ein Beispiel: Klasse "Ball"

Idee:
ein Ball ist rund und dient zum Spielen

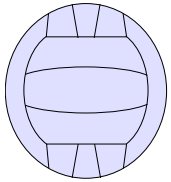


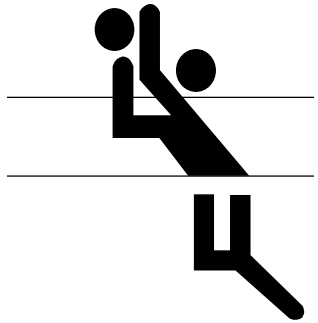
Attribute: Material, Größe, Gewicht, AktuellePosition

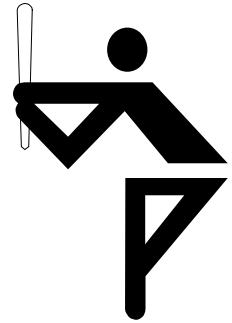
Funktionen: WoIstErDenn, BewegeNach

abgeleitete Klassen: Fußball Volleyball Baseball

Konkretisierung







Funktionen:

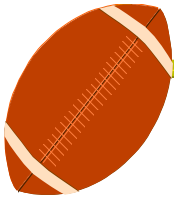
kicken
köpfen

pritschen
abblocken

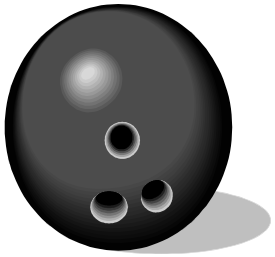
schlagen
fangen

Ideen sind vage, Begriffe sind unscharf

- ist ein Ball immer rund ?



- wieso heißt das runde Ding beim Kegeln nicht "Ball" ?

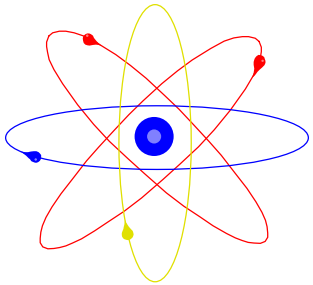


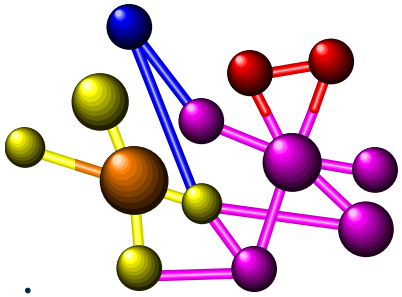
- ein Golfball ist doch auch hart !

- Begriffsbildung und Modellbildung hängen eng zusammen
- die Fähigkeit dazu ist notwendige Voraussetzung zur objektorientierten Programmierung und muß geübt werden

Kriterien für Modellierung (1)

- es geht nicht um "Korrektheit"
  - es gibt keine absolute Wahrheit
  - vgl. Modelle in Physik und Chemie





- sondern um Zweckmäßigkeit
  - welche Probleme kann ich mit einem Modell lösen
  - wie gut stimmt mein Modell mit der Wirklichkeit überein
  - Software: wie gut entsprechen die Funktionen den Anforderungen
- und einen großen Gültigkeitsbereich
  - nicht gerade Allgemeingültigkeit
  - aber möglichst Wiederverwendbarkeit

Kriterien für Modellierung (2)

ein Ball wird zum Fußball
mehr durch die Spielweise,
weniger durch seine Beschaffenheit

- zuerst die Funktionen !
- Attribute
  - an sich sind wenig interessant
  - eigentlicher Zweck: ermöglichen Realisierung von Funktionen
  - beschreiben nicht das "Wesen der Dinge",
    sondern die Freiheitsgrade
- zur Festlegung der Funktionen und
  dazu erforderlicher Attribute braucht es
  - klare Zielvorstellungen
    - was bezwecke ich mit dem Modell, wofür will ich es weiterverwenden ?
  - "Intelligenz, Erfahrung und guten Geschmack" (Stroustrup)

Nochmals - Grundprinzipien

- Kapselung
  - Objekteigenschaften (d.h. Daten) und Methoden bilden eine Einheit. Auf die Daten eines Objektes kann nur über seinen (öffentlichen) Methoden zugegriffen werden.
- Vererbung
  - Ein Objekt kann die Eigenschaften eines anderen Objektes erben. Eigenschaft ist dabei nicht nur Daten (Attribute) sondern auch alle Methoden.
- Polymorphie (Vielgestaltigkeit)
  - Objekte können auf die selbe Nachricht (Methode z.B. drucke()) unterschiedlich reagieren, und das erst zur Laufzeit. D.h. das Interface (die Schnittstelle) bleibt gleich aber die Methode ändert sich bei unterschiedlichen Objekten.