# Klassendiagramm:

**Klasse:** Sammlung von Exemplaren, Einschränkungen und Semantik. Können auch als Typ interpretiert werden.

**Objekt:** Ausprägung einer Klasse (Synonym: Instanz)

**Attribute:** repräsentieren Eigenschaften der Klasse und nicht des Objekts

Beispiel Klasse:

|  |
| --- |
| Partyteilnehmer*(Klassenname)* |
| - name: String *(Attribute)*  - wohnort: String  - geburtsdatum: Date |
| + feiern() *(Operation)* |

Beispiel Objekt:

|  |
| --- |
| Himbeer-Toni: Partyteilnehmer  *(Objektname) (Zugehörige Klasse)* |
| *(Attribute) (Typ und Wert)*  - name: String = ”Georg Mayer”  - wohnort: String = ”München”  - geburtsdatum: Date = ”1970-12-12“ |
| + feiern() |

## Notationselemente

**Klassenname:** Fettdruck, mittig, innerhalb eines Rechtecks, großgeschrieben.(Für abstrakte Klassen: kursiv)

**Abstrakte Klassen:** nicht instanzierbar! Kursiv bzw. mit zusatz {abstract}  
  
   
  
bzw

Feier  
{abstract}

*Feier*

**Abschnitte:** Um Attribute und/ oder Operationen darzustellen.  
**möglich:**zusätzliche Abschnitte im Attribute und/oder Operationen zu gruppieren

|  |
| --- |
| Partyteilnehmer |
| name : String  geburtsdatum : Date |
| << Datenbank>>  Database: String |
| <<Name>>  getName() : String  setName(neuerName : String) |
| <<Geburtsdatum>>  getGD() : Date  setGD(neuesGD : Date) |
| <<Persistenzoptionen>>  exportToXML(filename : String)  importFromXMML(filename : String)  writeToDB()  readfromDB() |

- Anzahl der zusätzlichen Abschnitte ist unbeschränkt  
- Abschnitte werden mit <<Name>> benannt.  
- Die Aufteilung ist abhängig von der Semantik.

**Attribut** üblicherweise im zweiten Abschnitt der Attributname wid:  
linksbündig,  
kleingeschrieben und  
unterstrichen wenn es ein Klassenattribut ist (static)

**Attributdeklaration:** [ Sichtbarkeit ]  
[ / ]  
name  
[ : Typ ]  
[ [ Multiplizität ] ]  
[ = Vorgabewert ]  
[ { Eigenschaftswert } [ , Eigenschaftswert ] \* } ]  
wobei:

**Sichtbarkeit:**

**public(+):** jedes andere Element hat uneingeschränkten Zugriff

**private(-):** Nur Instanzen der Klasse dürfen zugreifen.

**package(~):** für alle Elemente im selben Paket zugreifbar

**protected(#):** wie private aber auch Instanzen abgeleiteter Klassen

**/:** abgeleitetes Attribut, zur Laufzeit berechnet

**name:** kleingeschrieben, Sonderzeichen zulässig, auch mehrere Wörter

**Typ:** Datentyp des Attributs

**Multiplizität:** legt Unter- und Obergrenze der Anzahl ablegbare Instanzen fest. Optionales Attribut, höchstens darf ein Wert angegeben werden:

**1..1 = 1:** zwingendes Attribut, es wird genau ein Wert eingegeben

**0..\* = \*:** optional beliebig, beliebig viele

**1..\*:** beliebige Werte, aber mindestens einer

**n..m:** mindestens n, höchstens m Elemente

**Vorgabewert:** fester Wert, der automatisch gesetzt wird

**Eigenschaftswert:** Zum Benennen besonderer Charakteristiken des Attributs

**readonly:** für Konstanten

**ordered:** geordnete Reihenfolge der Inhalte

**unique:** Duplikate treten nicht auf

**Klassenattribut:** unterstrichen („statische“ Attribute in gängigen Programmiersprachen)

**Beispiel:** att1 : int  
+ att2 : int  
+ pi : double = 3.1415  
- att3 : boolean  
# att4 : short  
~ att5 : String = “Test” {readonly}  
 att6 : String[0..\*] {ordered}  
/ att7

Test1

Test2