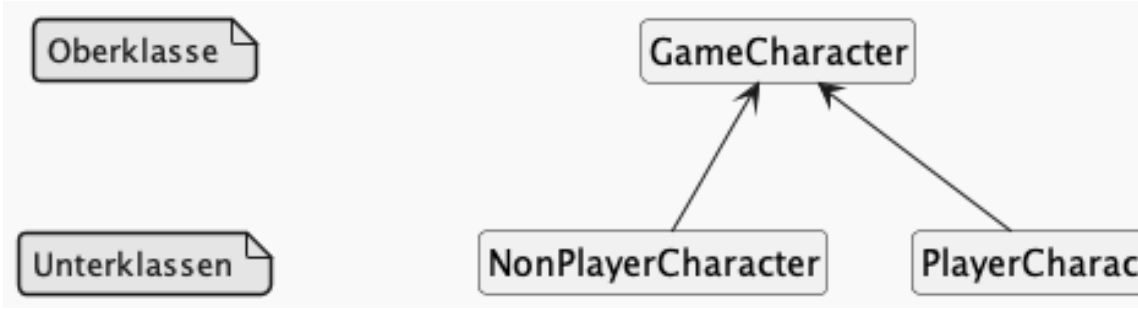
Vererbung

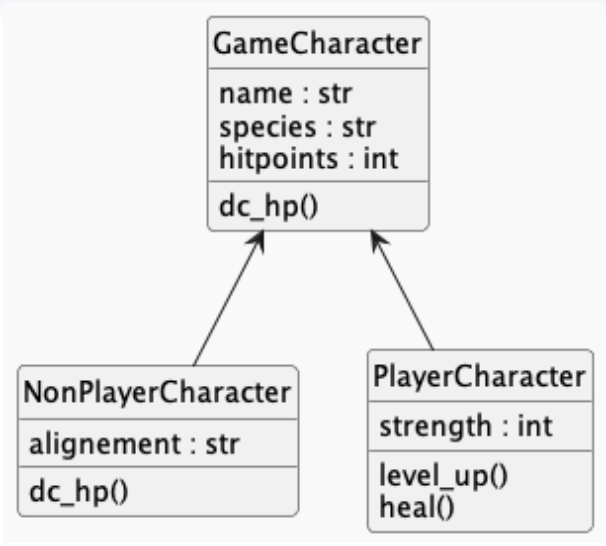

PALADIN (FEMALE)
FANTASY CLASSES
SERIES 1

- Es gibt Player Character und Non Player Character.
- Beides sind spezielle Charaktäre, welche gleiche Attribute und gleiche Methoden besitzen,
- aber auch unterschiedliche Attribute und Methoden aufweisen können

- Generalisierungshierarchie:



- Terminologie:
  - Oberklasse, Superklasse, Elternklasse, Basisklasse
  - Unterklasse, Subklasse, Kindklasse, abgeleitete Klasse.
- Unterklassen erben Atrribute und Methoden der Superklasse


GameCharacter
name : str
species : str
hitpoints : int
dc_hp()
NonPlayerCharacter
alignement : str
dc_hp()
PlayerCharacter
strength : int
level_up()
heal()

Beobachtung

- gleicher Programmcode wird bei unterschiedlichen Klassen benötigt.
- neues Abstraktionsverfahren wird benötigt: Vererbung
- Code wird an einer Stelle geschreiben, an einer anderen wieder verwendet

=> Fehlervermeidung

```
1  from dataclasses import dataclass
2  @dataclass
3  class GameCharacter:
4      name: str
5      species: str
6      hitpoints : int
7
8      def dec_hp(self, damage : int):
9          self.hitpoints = self.hitpoints - damage
10
11
```

```
@dataclass
class NonPlayerCharacter(GameCharacter):
    alignement : str

def dec_hp(self, damage : int = 2):
        self.hitpoints = self.hitpoints - damage
```

```python
@dataclass
class PlayerCharacter(GameCharacter):
    strength : int

    def levelup(self, percent : int):
        self.strength*(1+percent/100)

    def heal(self, healpoints):
        self.hitpoints = self.hitpoints + healpoints
```

- Durch die Vererbung kommen neue Attribute und Methoden hinzu.
- geerbte Methoden können dadurch überschrieben werden `dc_hp`, indem sie in der Subklasse neu definiert werden.
- Beim überschreiben der Methode müssen die Parameter der Methode übereinstimmen mit der Methode der Superklasse, kann aber mit weiteren Parametern ergänzt werden.
- es wird immer die Methode der Subklassen aufgerufen, wenn sie vorhanden ist.
- Beim Methodenaufruf können default-Werte angegeben werden. (`percent: int = 2`)