Unterschied zwischen einer Klasse und einem Objekt:

Klasse: Eine Blaupause oder ein Bauplan, der die Struktur und das Verhalten von Objekten beschreibt. Sie definiert, welche Attribute und Methoden die Objekte haben können.

Objekt: Eine Instanz einer Klasse. Es ist ein konkretes Exemplar, das auf der Klasse basiert und die darin definierten Eigenschaften und Methoden tatsächlich besitzt.

Was ist \_\_init\_\_ in Python?

\_\_init\_\_ ist der Initialisierungskonstruktor in Python. Es wird aufgerufen, wenn ein neues Objekt einer Klasse erstellt wird, und dient dazu, Attribute zu initialisieren.

python

Code kopieren

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

p = Person("Alice", 30) # \_\_init\_\_ wird aufgerufen

Unterschied zwischen einer Instanzvariable und einer Klassenvariable:

Instanzvariable: Gehört zu einem spezifischen Objekt und wird mit self definiert. Jede Instanz kann ihre eigenen Werte haben.

Klassenvariable: Gehört zur Klasse und ist für alle Instanzen gleich, solange sie nicht überschrieben wird. Sie wird direkt innerhalb der Klasse, aber außerhalb der Methoden definiert.

python

Code kopieren

class Beispiel:

klassen\_variable = 42 # Klassenvariable

def \_\_init\_\_(self, wert):

self.instanz\_variable = wert # Instanzvariable

Sichtbarkeitsmodifikatoren in Python im Vergleich zu Java:

Java: Sichtbarkeitsmodifikatoren wie public, private, protected.

Python: Es gibt keine strikten Modifikatoren. Sichtbarkeit wird durch Konvention geregelt:

\_name: "Protected", sollte als intern betrachtet werden.

\_\_name: "Private", wird durch Namensänderung (Name Mangling) unzugänglich gemacht.

Was kann man in Python tun, um zu "verstecken"?

Präfix mit \_\_ (doppelt unterstrichen) sorgt für Namensänderung (Name Mangling), um Konflikte zu vermeiden:

python

Code kopieren

class Beispiel:

def \_\_init\_\_(self):

self.\_\_versteckt = "secret"

b = Beispiel()

print(b.\_\_versteckt) # Fehler

print(b.\_Beispiel\_\_versteckt) # Zugriff möglich

\_\_str\_\_ und \_\_repr\_\_:

\_\_str\_\_: Gibt eine benutzerfreundliche, lesbare Darstellung des Objekts zurück (z. B. für print).

\_\_repr\_\_: Gibt eine repräsentative, möglichst eindeutige Darstellung zurück, die für Entwickler nützlich ist (z. B. in Debugging-Tools).

python

Code kopieren

class Beispiel:

def \_\_repr\_\_(self):

return "Beispiel()"

def \_\_str\_\_(self):

return "Dies ist ein Beispiel"

Klassenattribute:

Attribute, die zur Klasse und nicht zu einer Instanz gehören. Sie sind für alle Instanzen gemeinsam.

python

Code kopieren

class Beispiel:

klassen\_attribut = "Gemeinsam"

obj1 = Beispiel()

obj2 = Beispiel()

print(obj1.klassen\_attribut) # "Gemeinsam"

Beispiel.klassen\_attribut = "Geändert"

print(obj2.klassen\_attribut) # "Geändert"

Was passiert, wenn man ein Attribut auf ein Objekt schreibt, das nicht in der Klasse definiert ist?

Das Attribut wird dynamisch zum Objekt hinzugefügt und gehört nur zu diesem Objekt.

python

Code kopieren

class Beispiel:

pass

obj = Beispiel()

obj.neues\_attribut = 42 # Dynamische Zuweisung

print(obj.neues\_attribut) # 42

Was ist super() und wofür wird es verwendet?

super() ermöglicht den Zugriff auf Methoden oder Konstruktoren der Basisklasse aus einer abgeleiteten Klasse.

python

Code kopieren

class Parent:

def \_\_init\_\_(self):

print("Parent init")

class Child(Parent):

def \_\_init\_\_(self):

super().\_\_init\_\_() # Ruft Parent.\_\_init\_\_ auf

print("Child init")

@staticmethod:

Dekorator für Methoden, die keinen Zugriff auf die Instanz (self) oder die Klasse (cls) benötigen.

python

Code kopieren

class Beispiel:

@staticmethod

def hilfsmethode(x):

return x \*\* 2

Was ist Polymorphismus?

Die Fähigkeit, dass dieselbe Schnittstelle unterschiedliche Implementierungen haben kann.

In Python ist Polymorphismus durch Duck Typing üblich:

python

Code kopieren

class Katze:

def sprich(self):

return "Miau"

class Hund:

def sprich(self):

return "Wuff"

def tier\_spricht(tier):

print(tier.sprich())

tier\_spricht(Katze()) # "Miau"

tier\_spricht(Hund()) # "Wuff"

Mehrfachvererbung:

Eine Klasse kann von mehreren Basisklassen erben.

Vorsicht bei Konflikten, z. B. durch Diamond Problem:

python

Code kopieren

class A:

def sprich(self):

print("A")

class B(A):

pass

class C(A):

def sprich(self):

print("C")

class D(B, C):

pass

d = D()

d.sprich() # MRO entscheidet, welche Methode aufgerufen wird.

Was ist ein MRO (Method Resolution Order)?

Reihenfolge, in der Python nach Methoden oder Attributen sucht. Bestimmt durch C3 Linearization.

Kann mit ClassName.mro() oder help(ClassName) überprüft werden.

python

Code kopieren

print(D.mro()) # Zeigt Reihenfolge der Suche