Objektorientierte Programmierung (OOP)

Klassen in Python

4 Säulen der OOP

Generalisierung	Vererbung	Kapselung	Polymorphismus
Objekte klassifizieren und Gemeinsamkeiten feststellen	Hilft Fehler zu vermeiden und erlaubt Nutzung vorgefertigter Objekte und Klassen	Schutzmechanismus der Funktionalität bereitstellt, aber falsche Nutzung einschränkt	Überschreiben und Überladen von Methoden

Objekte aus der echten Welt → Objekte im Code

Eigenschaften eines Autos:

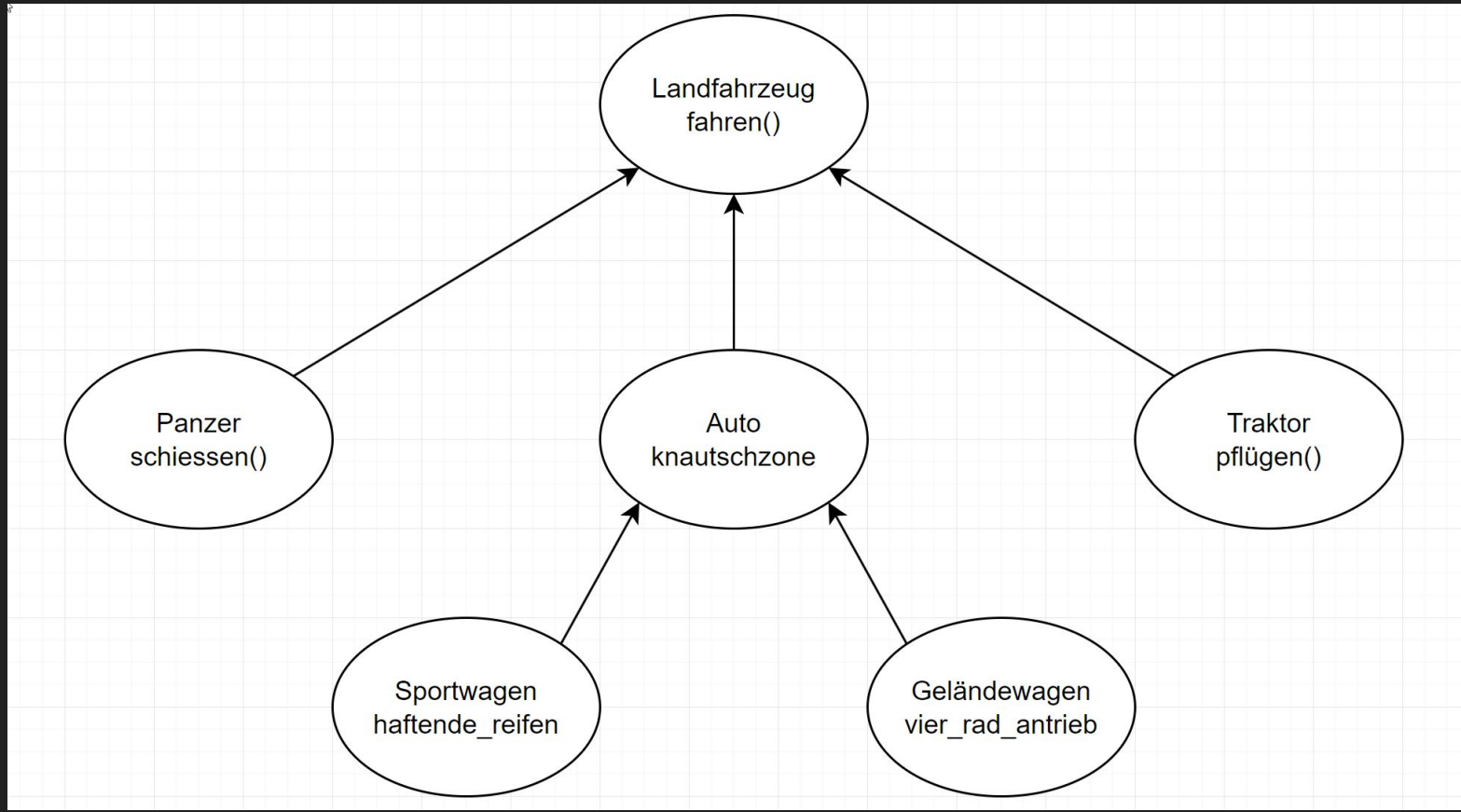
- Farbe
- Anzahl der Räder
- Marke (Modell)
- Lichter
- Anzahl der Sitze
- Motorisierung
- Baujahr

Objekte aus der echten Welt → Objekte im Code

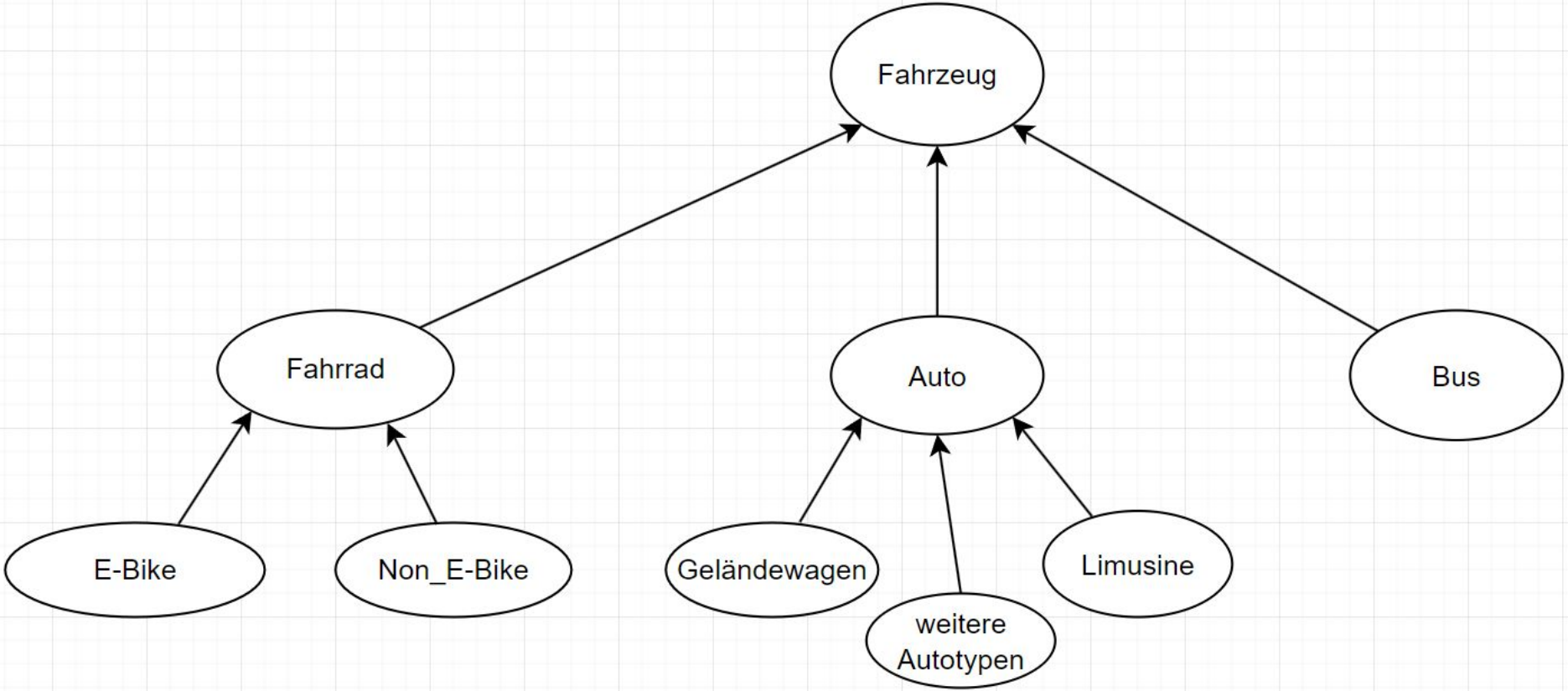
Funktionen eines Autos:

- fahren()
- auf_allrad_umstellen()
- rosten()

Vererbung. Diagramm und siehe Code



Alternatives Beispiel



4 Säulen der OOP

Generalisierung	Vererbung	Kapselung	Polymorphismus
Objekte klassifizieren und Gemeinsamkeiten feststellen	Hilft Fehler zu vermeiden und erlaubt Nutzung vorgefertigter Objekte und Klassen	Schutzmechanismus der Funktionalität bereitstellt, aber falsche Nutzung einschränkt	Überschreiben und Überladen von Methoden

4 Säulen der OOP

Generalisierung

Autos und Panzer haben eine **Farbe** und können **fahren()**

Vererbung

Autos sind auch Landfahrzeuge und haben die allgemeinen Eigenschaften und Funktionen aller Landfahrzeuge.

Farbe und **fahren()**

Kapselung

private Variablen

→

__Kilometerstand

Polymorphismus

Überschreiben von **fahren()**, sodass es nicht nur Landfahrzeug sondern Auto sagt