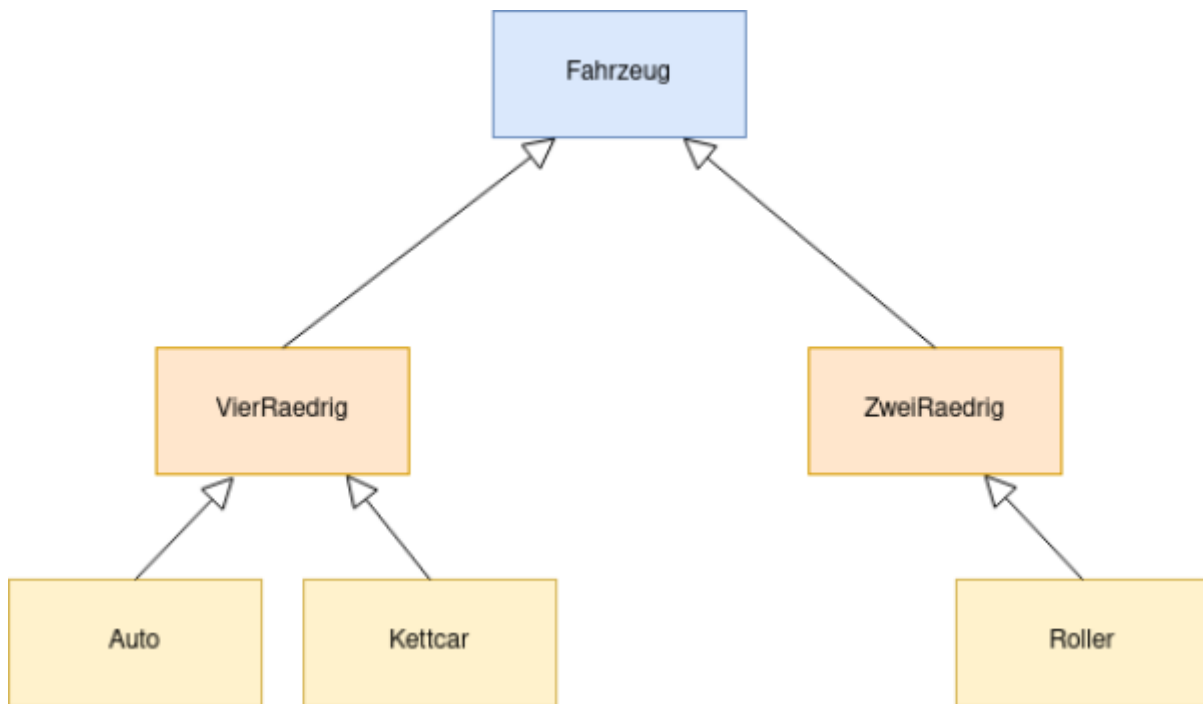


# Polymorphismus genauer

## Variablenpolymorphismus

**Polymorphismus** haben wir schon für Variablen kennengelernt: Eine Variable eines Supertyps kann auch Werte aller Subtypen halten - die Variable ist *polymorph*.



(A1)

Welche Typen können Werte haben, die in den folgenden Variablen gespeichert werden?

```
Fahrzeug f;  
Roller r;  
vierRaedrig v;
```

## Methodenpolymorphismus

### Problemstellung

Die Vererbungshierarchie unseres soziales Netzwerk mit Vererbung sieht gerade so aus:



Man sieht, dass die Methode zum Anzeigen eines Beitrags in der Klasse `Beitrag` definiert ist und an die Klassen `TextBeitrag` und `PhotoBeitrag` vererbt wird. Diese Methode weiß nichts über besondere Eigenschaften der Subklassen - Vererbung ist eine Einbahnstrasse. Das führt zum Problem, dass die Ausgabe aller Beiträge etwas so aussehen:

```
Leonardo da Vinci
40 seconds ago - 2 people like this.
No comments.

Alexander Graham Bell
12 minutes ago - 4 people like this.
No comments.
```

**TextBeitrag**

**PhotoBeitrag**

dabei werden die Besonderheiten der Beitragsarten nicht berücksichtigt - der `photoBeitrag` hat keine Bilddatei und keine Caption. Eigentlich sollte das nämlich so aussehen:

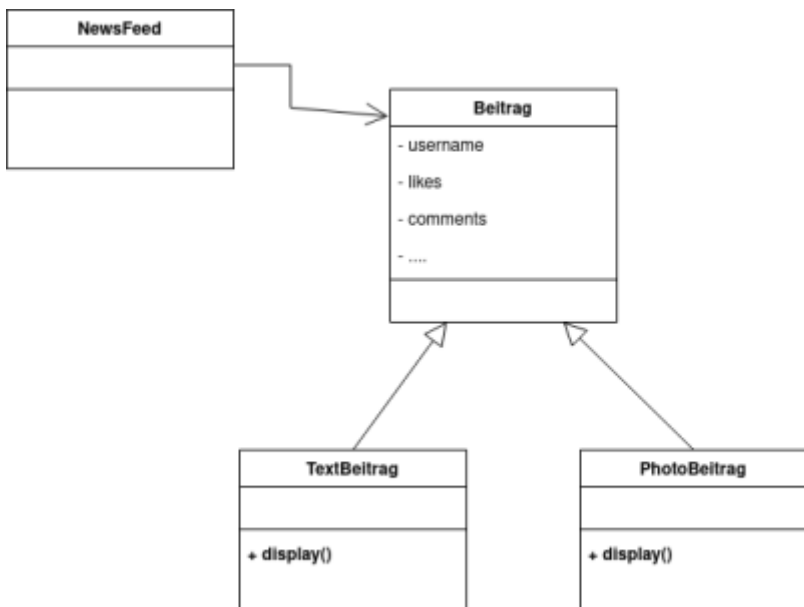
```

Leonardo da Vinci
Had a great idea this morning.
But now I forgot what it was. Something to do with flying ...
40 seconds ago - 2 people like this.
  No comments.
TextBeitrag

Alexander Graham Bell
[experiment.jpg]
I think I might call this thing 'telephone'.
12 minutes ago - 4 people like this.
  No comments.
PhotoBeitrag

```

Die spontane Lösungsidee verschiebt die `display`-Methode in die Subklassen, so dass jede Subklasse eine eigene `display`-Methode hat, welche dann natürlich entsprechend der spezifischen Eigenschaften implementiert sein könnte:



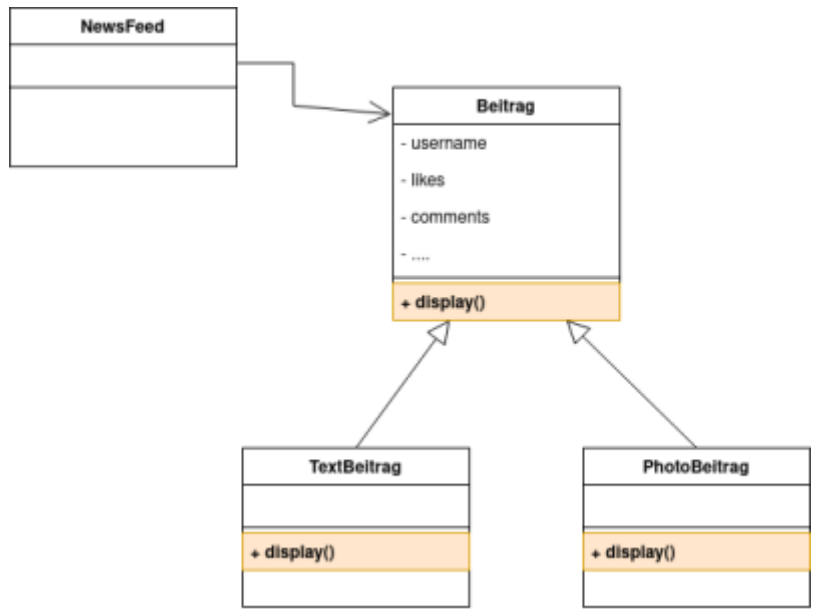
**Dieser Versuch ist zum Scheitern verurteilt:**

- Zugriff auf die privaten geerbten Attribute aus Beitrag ist nicht möglich.
- Die Klasse News feed benötigt eine `display`-Methode in Beitrag.

**Lösungsansatz: Überschreiben**

- Superklasse und Subklasse definieren Methoden mit gleicher Signatur.
- Jede der Methoden hat Zugriff auf alle Attribute (Felder) ihrer jeweiligen Klasse.
- Der Check des statischen Typs der Superklasse ist erfüllt.
- Die Methode der Subklasse wird erst zur Laufzeit aufgerufen und überschreibt dabei die Version der Superklasse.

Es gibt also `display`-Methoden in der Superklasse und in den Subklassen (wenn nötig):



Fragen:

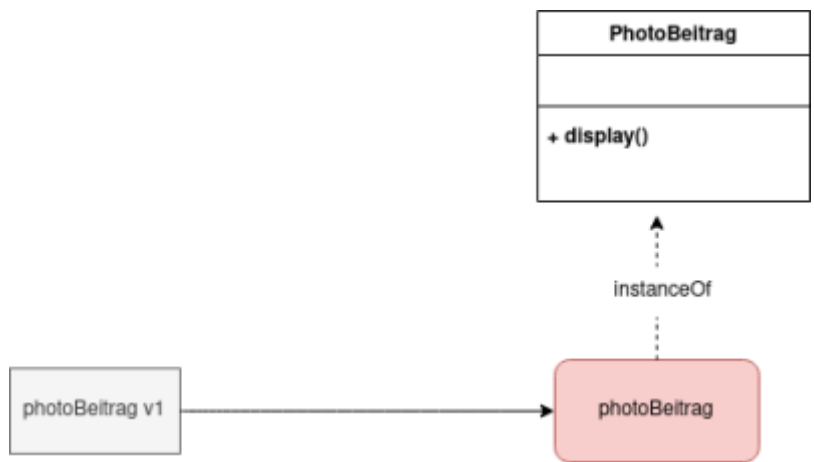
- Welche Rolle spielt die Version der Superklasse?
- Welche der `display`-Methoden wird denn zur Laufzeit tatsächlich aufgerufen?

## Methodenauswahl zur Laufzeit

### Ohne Vererbung

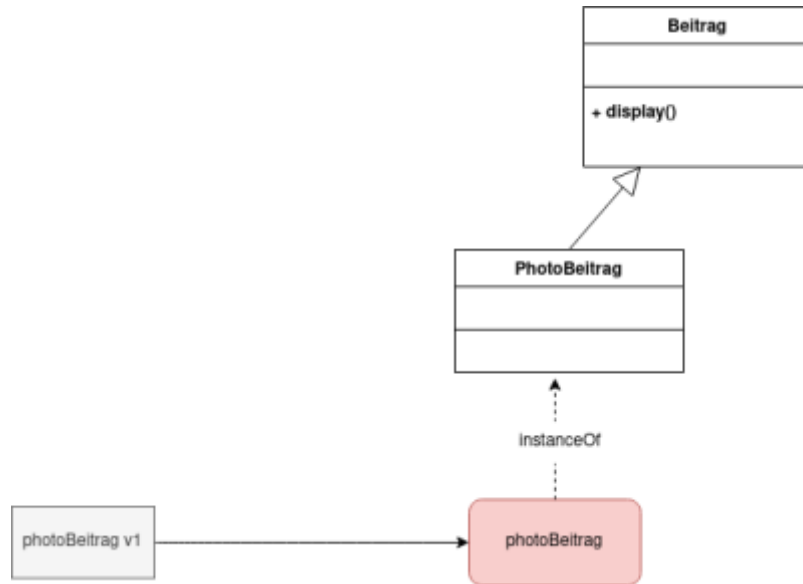
Keine Vererbung, kein Polymorphismus, kein Problem → die offensichtliche Methode wird ausgeführt:

```
v1.display()
```



### Vererbung, kein Überschreiben

```
v1.display()
```

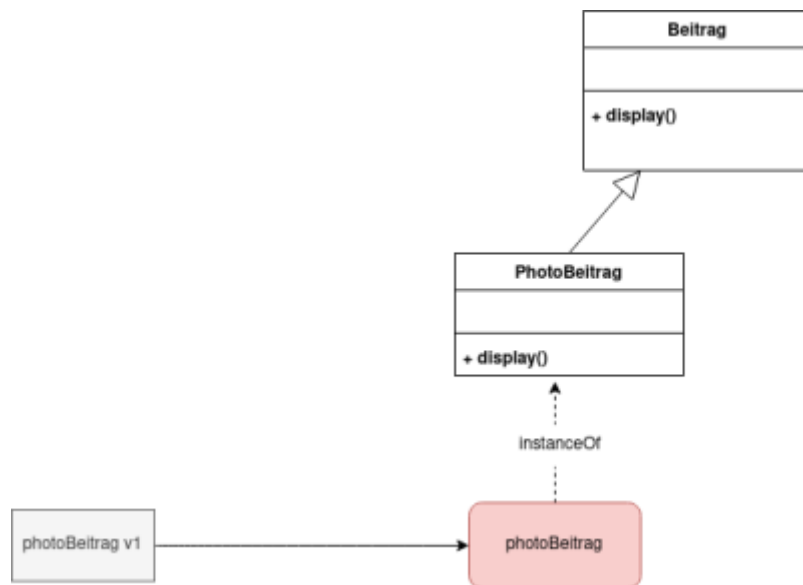


Bei der Suche nach der Methode wird die Vererbungshierarchie von unten nach oben durchlaufen (beginnend beim dynamischen Typ), bis zum Treffer - diese Methode wird ausgeführt.

Die Methode der Superklasse wird also solange von den Versionen der Subklasse(n) „verdeckt“, solange es in den Subklassen Methoden mit gleicher Signatur gibt.

### Vererbung und Überschreiben:

```
v1.display()
```



Bei der Suche nach der auszuführenden Methode wird die Vererbungshierarchie von unten nach oben durchlaufen (beginnend beim dynamischen Typ), **bis der erste Treffer gefunden wird** - diese Methode wird ausgeführt. Im Beispiel also die Methode, die in photoBeitrag implementiert ist.

# Material

<a href="#">auswahl_102.png</a>	26.4 KiB	29.11.2021	19:49
<a href="#">auswahl_103.png</a>	41.1 KiB	29.11.2021	19:49
<a href="#">fahrzeuge.drawio.png</a>	13.3 KiB	29.11.2021	14:55
<a href="#">kap11_polymorphie01.odp</a>	1.5 MiB	29.11.2021	19:46
<a href="#">kap11_polymorphie01.pdf</a>	360.2 KiB	29.11.2021	19:46
<a href="#">newssystem.drawio.png</a>	13.1 KiB	29.11.2021	15:06
<a href="#">newssystem01.drawio.png</a>	16.9 KiB	29.11.2021	19:52
<a href="#">newssystem02.drawio.png</a>	18.2 KiB	29.11.2021	19:56
<a href="#">poly01.png</a>	10.9 KiB	29.11.2021	19:59
<a href="#">poly02.png</a>	14.6 KiB	29.11.2021	19:59
<a href="#">poly03.png</a>	15.7 KiB	29.11.2021	19:59

From:  
<https://info-bw.de/> -

Permanent link:  
<https://info-bw.de/faecher:informatik:oberstufe:modellierung:vererbung:polymorphismus:start?rev=1638216293>

Last update: **29.11.2021 20:04**

