# Vererbung

## Vererbung - Modellierung

Die Vererbung ist ein grundlegender Ansatz in der objektorientierten Softwareentwicklung. Sie wird immer dann verwendet, wenn eine Klassenhierarchie gebildet werden kann. Grundprinzip der Vererbung ist, dass gemeinsame Strukturen zwischen Klassen (Attribute und Methoden) genutzt werden können, um Code zu sparen und Programme übersichtlicher zu gestalten.

|  |
| --- |
| Sparkonto |
| + kontonr: int  + inhaber: str  + kontostand: float  + zinssatz: float |
| + Sparkonto(p\_nr: int, p\_name: str)  + setZinssatz(p\_zinssatz: float): void  + einzahlen(p\_betrag: float): void  + auszahlen(p\_betrag: float): void  + zinsen\_gutschreiben(): void |

|  |
| --- |
| Girokonto |
| + kontonr: int  + inhaber: str  + kontostand: float  + dispo: float |
| + Girokonto(p\_nr: int, p\_name: str)  + setDispo(p\_betrag: float): void  + einzahlen(p\_betrag: float): void  + auszahlen(p\_betrag: float): void |

Sowohl ein Sparkonto als auch ein Girokonto sind Spezialisierungen eines Kontos. Bei einem Sparkonto wird das Guthaben zu einem bestimmten Zinssatz verzinst, d.h. die Zinsen werden dem Kontostand des Sparkontos gutgeschrieben. Das Girokonto kann von einem Kunden um einen Betrag *dispo* überzogen werden, d.h. das Konto hat einen Dispositionskredit. Folglich sind einige Attribute und Methoden sowohl beim Girokonto als auch beim Sparkonto gleich, andere unterscheiden sich.

|  |
| --- |
| Sparkonto |
| + kontonr: int  + kontoinhaber: str  + kontostand: float  + zinssatz: float |
| + Sparkonto(p\_nr: int, p\_name: str)  + setZinssatz(p\_zinssatz: float): void  + einzahlen(p\_betrag: float): void  + auszahlen(p\_betrag: float): void  + zinsen\_gutschreiben(): void |

|  |
| --- |
| Girokonto |
| + kontonr: int  + kontoinhaber: str  + kontostand: float  + dispo: float |
| + Girokonto(p\_nr: int, p\_name: str)  + setDispo(p\_betrag: float): void  + einzahlen(p\_betrag: float): void  + auszahlen(p\_betrag: float): void |

gleiches Attribut / gleiche Methode

unterschiedliches Attribut / unterschiedliche Methode

Möchte man wie bei der Vererbung üblich eine Klassenhierarchie bilden, könnte die Klasse *Konto* modelliert werden. Dabei besteht die Oberklasse *Konto* aus den gemeinsamen Attributen und Methoden der beiden „verwandten“ Unterklassen *Girokonto* und *Sparkonto*. Die Oberklasse *Konto* „vererbt“ dann die Attribute und Methoden an die Unterklassen *Girokonto* und *Sparkonto*, sodass diese dort nicht mehr modelliert werden müssen.

|  |  |
| --- | --- |
|  | Oberklasse /  Superklasse  Unterklassen |

Im dargestellten Beispiel erben die Unterklassen *Girokonto* und *Sparkonto* von der Oberklasse (Superklasse) *Konto* die Attribute *kontonr*, *kontoinhaber* und *kontostand*. Darüber hinaus werden auch die Methoden *einzahlen(...)* und *auszahlen(...)* vererbt. In den Unterklassen *Girokonto* und *Sparkonto* findet dann nur noch die Spezialisierung statt. So wird die Unterklasse *Girokonto* um das Attribut *dispo* und die Methode *set\_dispo(...)* erweitert. Die Unterklasse *Sparkonto* erhält im Vergleich zu Oberklasse Konto noch das Attribut *zinssatz* und die Methoden *set\_zinssatz(...)* und *zinsen\_gutschreiben()*.

Die Vererbung bewirkt, dass eine Klasse ihre Eigenschaften und Methoden an eine andere Klasse vererben kann. Die Vererbung wird im UML-Klassendiagramm durch einen Pfeil mit „leerer“ Spitze dargestellt.

**Allgemein**

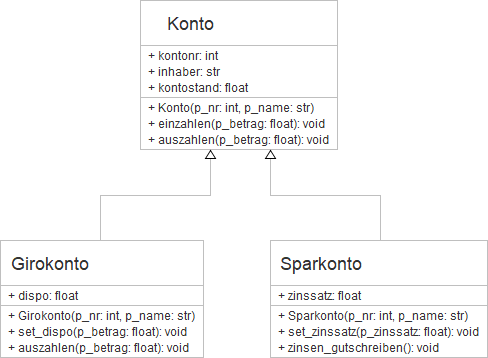
Das Grundprinzip der Vererbung besteht darin, aus bereits bekannten und beschriebenen Elementen eines Objektmodells weitere Elemente abzuleiten. Die Vererbung beschreibt somit eine Beziehung zwischen einer allgemeinen Klasse (Oberklasse, Superklasse) und einer spezialisierten Klasse (Unterklasse). Eine Unterklasse hat alle Attribute und Methoden der Oberklasse, ist aber um zusätzliche Attribute und Methoden erweitert und somit spezifiziert. Daher bezeichnet man Vererbungsstrukturen zwischen Klassen auch als *Generalisierung*/*Spezialisierung*.

|  |  |  |
| --- | --- | --- |
|  | A ist Generalisierung von B | B ist Spezialisierung von A |

Die Darstellung der Vererbungsstruktur im UML-Klassendiagramm erfolgt über einen Vererbungspfeil (mit transparenter Pfeilspitze), der von der erbenden Klasse (Unterklasse, Subklasse) zur vererbenden Klasse (Oberklasse, Superklasse) zeigt.

**Überschreiben von Methoden**

Eine Methode einer Oberklasse kann in der Unterklasse überschrieben werden. Dies ist immer dann sinnvoll, wenn in einer Unterklasse eine Methode anders ausgeführt werden soll als in der Oberklasse. Möchte man in der Unterklasse eine Methode aus der Oberklasse überschreiben, müssen sowohl der Methodenname als auch die Übergabeparameter gleich zur Methode der Oberklasse sein.



Ist die Methode *auszahlen(...)* in der Oberklasse *Konto* so implementiert, dass keine Überziehung des Kontos möglich ist, dann kann diese Methode für das *Sparkonto* übernommen werden. Das *Girokonto* kann allerdings in der Höhe des Dispositionskredits überzogen werden. Deswegen muss die Methode *auszahlen(...)* von der Unterklasse *Girokonto* überschrieben werden und taucht zusätzlich im Klassendiagramm der Unterklasse *Girokonto* auf.