1. Was ist ein Objekt?

Zuerst müssen wir unterscheiden, ob wir den Begriff aus einer allgemeinen, umgangssprachlichen Sicht nutzen oder eben im Umfeld der objektorientierten Programmierung (OOP).
Umgangssprachlich verstehen wir unter einem Objekt einen Gegenstand, also ein Haus, ein Auto, einen Pinsel, eine Türe usw. Diese Objekte können durch **Eigenschaften** beschrieben werden. So weist ein Auto z.B. eine Marke, einen Typ, eine Farbe usw. auf. Objekte können

- sehr unterschiedlicher Art sein.
- von der gleichen Art aber einer unterschiedlichen Ausprägung sein.
- in einer Hierarchie zueinander stehen, wobei gewisse Objekte real nicht existieren sondern nur der Verallgemeinerung eines Begriffes dienen.

Abb 1.1: verschiedene Objekte	*Abb 1.2:* Objekte gleicher Art	*Abb 1.3:* Objekte eines abstrakten Oberbegriffes (hier Säugetier)

In der OOP gehen wir nun aber einen Schritt weiter. „Unsere“ Objekte sollen ja etwas ausführen können, das heisst, dass sie selbständig funktionsfähig sind.
Ein Buch kann also z.B.

- seinen Titel, die Autorin, den Verlag usw. nennen
- sich öffnen und eine bestimmte Seite anzeigen
- den enthaltenen Text liefern
- usw.

Diese Denkweise ist fürs Erste sicher gewöhnungsbedürftig. Um Objekte in Software abzubilden, ist es aber unumgänglich, sich auf ein Rollenspiel einzulassen und sich gedanklich in die entsprechenden Objekte zu versetzen und so deren Aktivitäten und Zustände zu erkennen.
Objekt_(Programmierung)

Beispiel 1.1: Türe
Eine Türe verfügt über Eigenschaften wie Höhe, Breite, Farbe, Material.
Daneben kennt sie auch Zustände wie offen, geschlossen, verriegelt.
Um die verschiedenen Zustände zu erreichen, muss die Türe also über Fähigkeiten verfügen, so z.B. öffnen und schliessen. Für das ver- und entriegeln ist aber nicht die Türe sondern das Schloss zuständig. Diese Aufgaben werden also delegiert. Die **Delegation** ist eines der ganz wichtigen Prinzipien der OOP.
[*Das Prinzip der Delegation wird in der Learning Unit 5 vertieft behandelt.*]

Der oben beschriebene Sachverhalt lässt sich bezüglich der Zustände der Türe grafisch als Zustandsdiagramm darstellen. Es zeigt die **Zustände** sowie die nötigen **Effekte** (Fähigkeiten, Funktionen) auf, die zu einem Zustandswechsel (Transition) führen.

Abb 1.4: Zustandsdiagramm zu Türe
Zustandsdiagramm_(UML)

© René Probst

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/archiv/m320/learningunits/lu01/theorie/lu1-kapitel_1?rev=1788091600

Last update: **2026/08/30 14:06**