#10.1 Entities

Da wir jetzt wissen, wie man eigene Klassen erstellt und sie so nutzt, dass man komplexere Objekte damit erstellen kann, wollen wir die Möglichkeiten die Entities uns bieten nochmal genauer betrachten.

Eine Entity kann im Grunde jeder sichtbare Teil eines Spiels/Programms sein(und teilweise auch nicht sichtbare). Eine Spielfigur, eine Plattform, ein Projektil, ein Text, ein Hintergrund, ein grafischer Effekt, eine Lebensanzeige, ein Button, etc.

Eine Entity besteht im wesentlichen aus 2 Dingen.  
**Eigenschaften**(auch Felder genannt),

und **Methoden**.

## Eigenschaften

**Eigenschaften** beschreiben den IST-Zustand der Entity.  
Beispiel:

Wo sie ist, wie schnell sie ist, wie viel Leben sie hat, in welcher Animationsphase sie sich gerade befindet usw.

Die **Eigenschaften** bestehen aus Variablen

Beispiel:

Vector2f position, für die Position

Vector2f speed, für die Geschwindigkeit

int life, für die Lebenspunkte

Wie alle Variablen, kann man die **Eigenschaften** einer Entity lesen oder ändern. Will man die Entity bewegen, verändert man ihre Positionsvariable.

## Methoden

**Methoden** beschreiben anders als Eigenschaften nicht den IST-Zustand, sondern wie sich die Entity verhält. Sie sind ausführbare Teile von Code. Man gibt ihnen beim ausführen Parameter und sie geben einen Rückgabewert zurück. Dies entspricht der Mathematischen Sichtweise von Funktionen(z.b. f(x) = x²), aber in der Informatik benutzt man sie auch für Objekte, also auch unsere Entities.

Immer wenn die Entity etwas tut, muss das in einer **Methode** stehen. Man kann sich also vorstellen, die **Methoden** sind wie alle Handlungen einer Entity. Diese sind aufgrund ihrer Funktionsweise oft vom Rückgabetyp "void".

Beispiel

Man möchte eine Entity erstellen, die einem Auto ähnelt. Was kann man mit einem Auto machen? Es fahren. Also würde man in die Klasse eine "fahren" **Methode** schreiben. Die Parameter wären die Informationen die man braucht, wenn man fährt. Das wäre hier z.B. das Fahrziel.

Oder man möchte eine Wecker-Entity erstellen. Was kann man mit einem Wecker machen? Ihn stellen. Also eine "weckerStellen" **Methode**, die nötigen Parameter wäre die Uhrzeit, auf die man sie stellt. Ein Wecker kann auch klingeln. Dafür könnte man eine "klingeln" **Methode** machen. Parameter braucht man hier allerdings keine.

## Objekt Orientierung

Was man damit erreichen möchte ist, dass alle Dinge, die eine Entity betreffen, also ihre Eigenschaften, wie sie sich verhält und was man mit ihr machen kann, in nur eine Code-Klasse stehen. Man versucht alles in seinem Programm in Objekte aufzuspalten, dies nennt sich Objekt Orientierung.

Die wirklichen Vorteile der Objekt Orientierung sind Verberbung, Polymorphismus und Kapselung. Diese werden wir aber erst in später benutzen.