AB06 - Attribute

Hühner sind schlauer als viele vermuten. Besonders wenn DU die Hühner programmierst. Dann können sie sich sogar dauerhaft etwas merken und Fragen dazu beantworten…

Die Figuren merken sich was dauerhaft …

> **Ziel:** Attribute als Speicher für jeweils einen Wert kennen und einsetzen können. Den Konstruktor einer Klasse kennenlernen und zum Initialisieren von Attributen verwenden können.

Attribute beschreiben Eigenschaften und Zustände

Wir sehen die Figuren in ihrer Welt agieren. Wir sehen auch, wie viel Hunger sie haben. Mit `getHunger()` können wir das auch abfragen. Dann erhalten wir von der Figur eine Zahl als Antwort. Woher weiß die Figur aber, wie viel Energie sie noch hat? Wie merkt sie sich diese Zahl?

Natürlich mit einer Variablen. Die hast du ja schon in Scratch kennengelernt. Jedes Objekt verwaltet dabei seine Variablen selbst. Wenn also mehrere Figuren auf der Welt herum laufen, dann kann jede einen unterschiedlichen Wert für Hunger speichern. Diese Variablen, in denen die Objekte etwas dauerhaft speichern, nennt man Eigenschaften oder Attribute des Objekts. Daneben gibt es auch lokale Variablen, die nur innerhalb einer Methode verwendet werden und danach wieder vergessen werden, und Parameter, die an eine Methode übergebene Werte speichern.

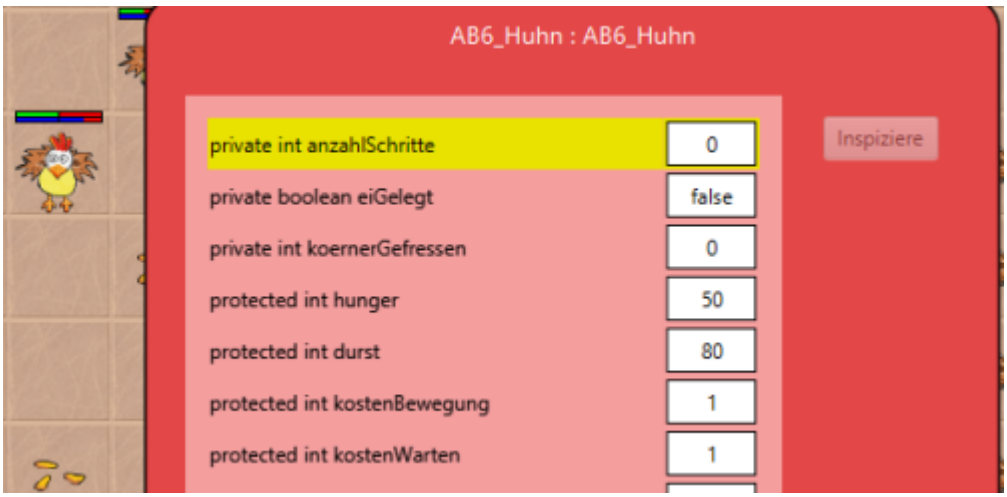
Den aktuellen Wert von Eigenschaften kann man in der Regel mit einer Methode abfragen, deren Name mit "get…" oder "gib…" beginnt, z.B.:

- `getHunger()` bzw. `getDurst()` wird dir die Restenergie als Antwort nennen

Die aktuellen Werte aller Attribute bestimmen den **Zustand** einer Figur.

(A1)

Rufe bei zwei Hühner das Inspect-Fenster auf (Rechtsmausklick auf das Huhn → Inspizieren) und vergleiche den Wert der Attribute `hunger` und `durst` mit der Lebensanzeige des Huhns. Lasse das Huhn laufen und beobachte die Werte der Attribute. Rechts siehst du ein Huhn und sein INSPECT-Fenster.



(A2)

Notiere, welche weiteren Eigenschaften des Huhns in Attributen gespeichert sind. Nenne mindestens eine Methode, mit der Du den Wert eines Attributs ändern kannst.

Wie Variablen funktionieren

Jede Variable zeigt im Hintergrund (für uns nicht sichtbar) auf genau eine Stelle (= Adresse) im Arbeitsspeicher des Computers. Damit wir uns nicht die kryptische Adresse merken müssen gibt es die Variablennamen. Mithilfe des Variablennamens können wir uns jederzeit den Wert anzeigen lassen der an dieser Stelle gespeichert wird oder ihn auch ändern. Namen von Variablen beginnen mit einem Kleinbuchstaben. Setzt sich der Name aus mehreren Wörtern zusammen, darf man keine Leerzeichen verwenden, sondern man beginnt jedes neue Wort mit einem Großbuchstaben. Das bezeichnet man als Camel-Case. Aus „Schritte bis zur Wand“ wird also z. B. „schritteBisZurWand“.

An jeder Adresse gibt es zu jedem Zeitpunkt genau einen aktuell gültigen Wert. Mit set-Methoden (und auch durch andere Methoden) werden die Werte geändert. Mit der get-Methode wird nach dem aktuellen Wert gefragt und dieser Wert wird dann dem Anfragenden als Ergebnis genannt. Dabei verbleibt der Wert unverändert in der Variable.

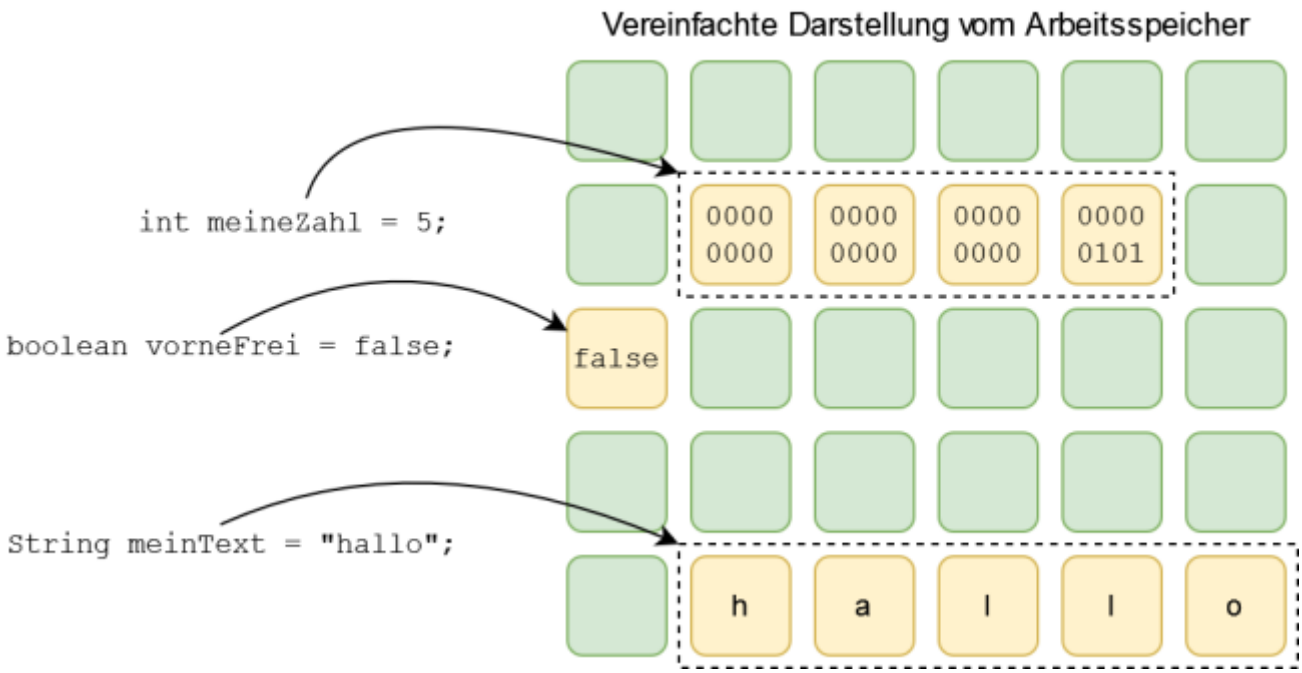
Der Typ der Variable legt u. a. die Größe des Speicherplatzes fest und außerdem sagt er dem Programm, als was es die Daten interpretieren muss. Du weißt bereits, dass Computer Daten als Binärzahlen speichern. Wenn man nun an eine Speicherplatz im Arbeitsspeicher schaut, dann wird man nur lange Folgen aus 1 und 0 sehen ohne zu wissen, ob man diese z. B. gerade als Zahl, als Buchstaben oder als etwas gänzliches anderes interpretieren muss. Nur durch den Datentyp weiß der Computer z. B.: "Aha, diese Bitfolge muss ich als Ganzzahl (int) interpretieren!".

Für uns relevant sind zunächst die folgenden drei Datentypen:

- **int** (32 Bit = 4 Byte groß) für Ganzzahlen zwischen -2.147.483.648 und +2.147.483.647

- **boolean** (1 Byte) für den Wahrheitswert true oder false
- **String** (flexible Länge) für beliebige Texte

Innerhalb des Computers kann man sich das alles dann wie folgt vorstellen:



Die Zeichnung zeigt drei Variablen unterschiedlichen Typs. Für den Integer-Wert wurde direkt die Binärdarstellung gewählt, da du diese bereits kennst (101_2 ist 5_{10}, vorne wird mit 0ern aufgefüllt). Boolean und String werden hier im Arbeitsspeicher vereinfacht dargestellt (sind „in echt“ auch Binärzahlen).

Das Programm weiß dank des Datentyps also sowohl, in wie vielen Bytes an Arbeitsspeicher es nachschauen muss und auch, wie es die dortigen 0er und 1er zu interpretieren hat.

From:
https://www.info-bw.de/ -

Permanent link:
https://www.info-bw.de/faecher:informatik:mittelstufe:bauernhof:ab6:start?rev=1718704579

Last update: **18.06.2024 09:56**