Umgang mit Variablen

Description

In so gut wie jedem Programmier-Tutorial geht es, meist beiläufig, um Variablen. Heute möchten wir unsere Aufmerksamkeit den wichtigsten Bestandteilen der Programmierung widmen.

Was ist eine Variable?

Bei einer Variable handelt es sich um einen Behälter, der einen Wert enthält. Dabei kann dieser Wert eine Zahl sein, muss es aber nicht. Es kann sich auch um einen String handeln, einer Formel oder ein Objekt. Der Behälter ist somit ein Speicherort, in den wir Informationen ablegen, die später gebraucht und ggf. verarbeitet werden.

Beispiele für Variablen

Wie gesagt, kann eine Variable sehr vielfältig sein. Hier ein paar Beispiele, wie Variablen aussehen können.

```
schalter = false;
x = 102;
wert = 87.57;
test = 'Hallo Welt';
objekt = instance_create(x, y, obj_feind);
farbe1 = c_white;
yy = 10 + (i * abstand);
array[0] = 42;
```

Alles, was vor dem **=** Zeichen ist, ist die Variable. Alles was folgt, ist das, was wir in dieses Gefäß rein tun. Wir sehen, dass eine Variable eine ganze Zahl, eine Zahl mit Nachkommastellen, ein Text, ein Farbwert, ein Objekt, ein Feld und eine Gleichung, sogar mit dem Inhalt einer weiteren Variable, sein kann.

Bezeichnung von Variablen

In jeder Programmiersprache gibt es gewisse Regeln, wie die Variablen bezeichnet werden dürfen. Im GameMaker muss man zusätzlich wissen, dass jedes Objekt von Haus aus bereits viele Variablen mitliefert, wie zum Beispiel *x*, *y*, *visible*, *speed* und vieles mehr. Wenn man diese in den Code schreibt, werden sie automatisch rot. Wird das Wort orange, handelt es sich um einen Befehl. Wenn es grau bleibt, kann man es als eigene Variable verwenden.

In manchen Programmiersprachen, wie etwa PHP, müssen Variablen mit einem Dollarzeichen beginnen. Im GameMaker ist das nicht der Fall.

Eine Variable darf keine Leerzeichen enthalten. Wenn sie aus mehreren Worten besteht, sollte man das folgende Wort mit einem Großbuchstaben beginnen, oder die Worte durch einen Unterstich trennen. Beispiel: *istInBewegung* oder *ist_in_bewegung*.

Eine Variable darf nur aus Buchstaben und Ziffern bestehen. Das erste Zeichen muss immer ein Buchstabe sein. Groß- und Kleinschreibung sind erlaubt und es wird auch unterschieden. *IstInBewegung* ist eine andere Variable als *IstInBewegung*.

Variablen dürfen keine deutschen Umlaute oder ein ß enthalten.

Als einziges Sonderzeichen ist ein Unterstrich erlaubt. Beispiel: *ist_in_bewegung*

Eine Variable darf nicht mit einem reservierten Wort (wie oben beschrieben, ein Befehl), identisch sein.

Die maximale Länge sind in GameMaker 64 Zeichen. Das gilt für den Namen, nicht für den Inhalt.

Variablen deklarieren

In manchen Programmiersprachen muss man erst einmal eine Variable definieren und bestimmen, um was für einen Typ es sich überhaupt handelt. Das kann dann wie folgt aussehen:

```
int x = 0;
```

int sagt aus, dass es sich um einen Integer handelt. Das ist ein Datentyp, der eine ganzzahlige Zahl beinhaltet. Diese Definitionen werden verwendet, um den richtigen Speicher zu reservieren. Neben int gibt es noch weitere Datentypen, wie *long*, *long int*, *byte*, *boolean*, *char*, *double* und so weiter. Je nach Programmiersprache kann sich das unterscheiden. In GameMaker braucht man das nicht!

Bevor man eine Variable verwendet, muss man aber vorher sagen, dass es sie gibt. Man sagt dem Programm, dass ein Gefäß existiert, der verwendet werden kann. Am einfachsten schreibt man den Startwert der Variable in das **Create-Event** des entsprechenden Objekt oder am Anfang eines Scriptes.

```
nameDerVariable = 0;
```

Ab jetzt kann man die Variable *nameDerVariable* im entsprechenden Objekt bzw. Script verwenden. Wenn man eine Variable im GameMaker nur an einer bestimmten Stelle, wie einem Script, braucht, ist

es besser, sie vorher mit **var** zu definieren. Das schränkt die Reichweite ein und spart u. A. Speicher. Beispiel:

```
var nameDerVariable;
```

Man kann mit dieser Methode auch mehrere Variablen in einer Zeile definieren.

```
var nameDerVariable, xx, yy;
```

Rechenoperationen mit Variablen

Das Thema Variablen, Zahlen und Gleichungen möchten wir nun ein wenig vertiefen. Es gibt unterschiedliche Methoden, Rechenoperationen durchzuführen. Hier ein Beispiel:

```
wert1 = 50;
wert2 = 150;
ergebnis = wert1 + wert2;
```

Wir definieren erst die Variablen *wert1* und *wert2*. Bei der Definition von *ergebnis* zählen wir die beiden Werte zusammen. Im konkreten Beispiel könnte es auch so lauten:

```
ergebnis = 50 + 150;
```

oder

```
ergebnis = 200;
```

Mit den Variablen sind wir aber flexibler und können Daten verarbeiten, die beispielsweise der User eingibt. Natürlich funktionieren nicht nur Additionen, sondern alle Grundrechenarten, Pi, Wurzeln und vieles mehr. Man kann darüber hinaus auch Klammern verwenden. Hier mal eine etwas komplexere Gleichung:

```
r = (128 + (spectrum) * sin(colour * pi / 32)) / 256;
```

Hinweis: Ganze Zahlen werden im GameMaker üblicherweise ohne Komma angegeben. Wir schreiben 128 und nicht 128.0. Bei der Shaderprogrammierung ist das anders. Hier muss 128.0 geschrieben werden, sonst erhalten wir eine Fehlermeldung.

Zahlen kann man auch leicht Werte hinzufügen. *ergebnis*++ sagt aus, dass der Variable *ergebnis* eine Zahl addiert wird. Aus ehemals 200 wird dann 201. Bei *ergebnis* += *10* wird dem Ergebnis der Wert 10 addiert. Das ist eine vereinfachte Schreibweise und bedeutet das Gleiche wie *ergebnis* = *ergebnis* + *10*.

Zeichenketten mit Variablen

Eine Variable kann auch einen Text enthalten. Damit kann man zwar nicht rechnen, aber dennoch kann dies sehr nützlich sein. Beispielsweise kann der Spieler seinen Namen eintragen und dieser Name wird im Spiel angezeigt. Oder wir lagern unsere Texte in Variablen aus und rufen dann nur noch die Variable an der entsprechenden Codestelle aus.

```
text = 'Das ist ein Text';
draw_set_color(c_white);
draw_text(20, 20, text);
```

Wenn man das im GameMaker im **Draw-Event** anwendet, erscheint auf dem Bildschirm „Das ist ein Text“. Ohne die Variable würde der Code so aussehen:

```
draw_set_color(c_white);
draw_text(20, 20, 'Das ist ein Text');
```

Man kann auch Zahlen als Variable anzeigen und dies sogar kombinieren.

```
text = 'Deine Punktzahl: ';
punkte = 15000;

draw_set_color(c_white);
draw_text(20, 20, text + string(punkte));
```

Auf dem Bildschirm erscheint nun „Deine Punktzahl: 15000“. Das können wir auch weiter ausbauen.

```
punkte = 15000;
bonus = 1700;
faktor = 3;

summe = punkte + bonus * faktor;

text1 = 'Gratuliere! ';
text2 = 'Du hast ';
text3 = ' Punkte und ';
text4 = ' Bonuspunkte! ';
text5 = 'Mit einem Bonusfaktor von ';
text6 = ' hast du insgesamt ';
text7 = ' Punkte erreicht!';

draw_set_color(c_white);
draw_text(20, 20, text1 + text2 + string(punkte) + text3 + string(bonus) + tex
```

Variablen in anderen Objekten

Jedes Objekt hat seine eigenen Variablen, aber man kann auch auf die Variablen anderer Objekte zugreifen. Um diese abzufragen, muss man lediglich den Namen des Objekts vor die Variable schreiben und mit einem Punkt trennen. Wenn wir den x-Wert des Gegners ermitteln wollen, greifen wir mit: *obj_gegner.x* darauf zu. So können wir auch Werte von Objekten verändern. Hier ein konkretes Beispiel:

```
gegner = instance_create(100, 100, obj_gegner);
gegner.speed = 2;
gegner.direction = 0;
```

Wir erstellen eine Instanz des Objekts obj_gegner und merken sie uns in der Variable *gegner*. Mit dieser Variable können wir nun auf die Variablen der erstellten Instanz zugreifen. Im Beispiel stellenwir die Geschwindigkeit auf 2 und senden den Gegner nach rechts.

Globale Variablen

Globale Variablen sind eine feine Sache. Man kann von jedem Script und jedem Objekt heraus darauf zugreifen. Globale Variablen definiert man mit dem Zusatz *global.* vor der Variable.

```
global.musikAn = true;
```

Jetzt kann man von überall aus diese Variable abfragen. Das ist von Vorteil, wenn man Einstellungen definiert, die für das ganze Spiel gelten. Musik, Soundeffekte, Lautstärke, Partikel, Blut und vieles mehr. Alles, was man in mehreren Räumen aus verschiedenen Objekten heraus abfragt, kann als globale Variable sinnvoll sein.

Solche Variablen sind aber mit Vorsicht zu genießen. Da diese Variablen immer definiert, abgefragt und geändert werden können, kann es sehr schnell zu Problemen kommen, wenn man versehentlich für zwei unterschiedliche Dinge den gleichen Namen verwendet. *global.SoundAn* kann sich auf die Soundeffekte beziehen, oder auf Soundeffekte und Musik. Wenn man nicht aufpasst, bekommt man bei den selbst programmierten Optionen Fehler rein und sucht dann ganz schön lange nach der Ursache. GameMaker gibt leider keine Auskunft darüber, ob eine Variable bereits existiert, weder lokal noch global!

Um dieses Problem in den Griff zu bekommen, ist es ratsam, alle globalen Variablen in einem Script zu definieren. Man legt sich ein Script an, welches zum Beispiel *scr_globaleVariablen* heißt. Dort definiert man jede globale Variable, die im Spiel auftreten kann. Jede, ohne Ausnahme! Nun hat man eine zentrale Stelle und kann an einem Ort nachschauen, ob es die Variable bereits gibt oder nicht. Das Script wird anschließend im ersten Raum des Spiels als erstes im **Create-Event** aufgerufen. Meistens hat man vor dem ersten Level ein Menü und vor dem Menü noch einen Ladebildschirm oder das Logo des Spiels. Da muss bereits das Script aufgerufen werden. Ab dann hat man ein entspanntes Leben.

Date Created
17. Oktober 2016
Author
sven