



Arbeiten mit Zufallszahlen

Description

Wie ich in meinem [letzten Gamedesign-Artikel](#) schrieb, kann man mit zufälligen Ereignissen zwar viel Unheil anrichten, aber auch tolle Sachen veranstalten. Heute wollen wir uns die praktische Umsetzung im GameMaker anschauen.

Wozu braucht man Zufallszahlen?

Um Zufallszahlen zu erzeugen, kennt der GameMaker mehrere Befehle. Mit diesen Zufallszahlen lassen sich verschiedene Dinge veranstalten. Zum Beispiel kann man damit einen Würfel simulieren, oder einen Effekt zufällig variieren, damit er nicht immer gleich aussieht. Wenn der Spieler ein Objekt einsammelt und an dem Platz die Punktzahl erscheint, kann man die Punktzahl mit einer zufälligen Geschwindigkeit ausblenden lassen, damit es nicht komisch wirkt, wenn der Spieler mehrere Objekte zur gleichen Zeit eingesammelt hat. Gleiches gilt auch für das Startframe einer Animation. Manch ein Spieler kennt das von FIFA und vergleichbaren Spielen. Nach einer bestimmten Aktion reagieren mehrere Figuren identisch und zeitgleich. Das sieht extrem komisch aus und dies kann man mit zufälligen Variablen, wenn man sie richtig setzt, in den Griff bekommen.

Natürlich gibt es auch die Möglichkeit, eine Figur an einer zufälligen Position erscheinen zu lassen oder mit einer Figur an eine zufällige Position im Spiel zu springen.

Du kannst Dir leicht vorstellen, dass es viele Gelegenheiten gibt, in denen eine Zufallszahl helfen kann. Um ein praktisches Beispiel zu zeigen, werden wir einen zufälligen Bot-Namen aus einer Liste wählen. Es wäre dabei hilfreich, wenn Du Dich bereits mit der [switch-Anweisung](#) auskennen würdest.

Zufallsgenerator starten

Ein Zufallsgenerator beginnt meist am selben Ausgangspunkt. Das bedeutet, dass die erste erzeugte Zahl immer gleich ist. Da wir aber eine echte Zufallszahl möchten, müssen wir den Zufallsgenerator einmal starten, bevor wir die Zufallsfunktionen verwenden. In GameMaker macht man das mit folgendem Befehl:

```
randomize();
```

Wenn wir das nicht machen, wundern wir uns später, warum, beispielsweise, der erste Bot immer Paul heißt und die anderen unterschiedlich sind.

Der random Befehl

Mit *random(wert)* erzeugen wir eine zufällige Zahl. Diese Zahl muss in Klammern stehen. Die erzeugte Zahl hat immer zwei Stellen hinter dem Komma und liegt zwischen 0 und dem vorgegebenen Wert. Wenn wir *random(6)* eingeben, liegt der Wert zwischen 0 und 6.

Diese Information beinhaltet zwei Aussagen. Erstens: mit *random(1) * 100* kann man eine Zahl erzeugen, die zwischen 0 und 100 liegt. Natürlich enthält auch diese Zahl wieder zwei Stellen hinter dem Komma. Wenn wir also ganze Zahlen haben wollen – und das ist die zweite Aussage, müssen wir runden.

```
zufallsZahl = round(random(1) * 100);
```

Mit *round()* werden Zahlen gerundet. Beachte, dass der round-Befehl auch die 100 einschließt. *round(random(1) * 100)* ergibt immer 100 oder 0, weil nur *random(1)* gerundet wird.

Es gibt in GameMaker noch andere Methoden, um eine Zahl zu runden. Mit *floor()* wird immer abgerundet, also nur die Zahl vor dem Komma ausgegeben. Mit *frac()* erhält man nur die Zahl nach dem Komma zurück und mit *ceil()* bekommen wir die nächst höhere Zahl.

Um das zu veranschaulichen, erzeugen wir einen Wert für einen Würfel. Hierbei brauchen wir eine Zahl zwischen 1 und 6. Mit *random()* gibt es nun zwei Ansätze.

```
// entweder  
zufallsZahl = floor(random(6) + 1);
```

```
// oder  
zufallsZahl = ceil(random(6));
```

Du kannst beide Methoden verwenden, das Ergebnis ist identisch.

Der random_range Befehl

Es kann die Situation entstehen, dass man eine zufällige Zahl haben möchte, die zwischen zwei Zahlen liegt. Das Problem kann man auch mit *random()* lösen, *random_range()* ist aber eleganter. Auch hier haben wir das Problem, mit den Nachkommastellen, können dies aber auf dieselbe Weise lösen wie bei *random()*. Nehmen wir an, wir möchten eine Zahl zwischen 4 und 10 erzeugen:

```
zufallsZahl = random_range(4, 10);
```

Nun erhalten wir entsprechende Werte zwischen 4 und 10. Um auf das konkrete Beispiel mit dem Würfel zurück zu kommen:

```
zufallsZahl = floor(random_range(1, 7));
```

Sieben? Warum Sieben? Nun, wir runden immer ab. Die höchste erzeugte Zahl wird 6.99 sein, also abgerundet 6 ergeben.

Zufallsnamen

Nun wollen wir endlich die Namen realisieren. Dazu erzeugen wir eine zufällige Zahl und legen per switch-Anweisung fest, was mit dieser Zahl passieren soll. Das Ergebnis zeigen wir im Draw-Event an.

Event Create:

```
randomize();
zufallsZahl = floor(random(10));

switch (zufallsZahl)
{
    case 0:
        botName = 'Paul';
        break;
    case 1:
        botName = 'Lisa';
        break;
    case 2:
        botName = 'Klaus';
        break;
    case 3:
        botName = 'Gustav';
        break;
    case 4:
        botName = 'Joel';
        break;
    case 5:
        botName = 'Anna';
        break;
    case 6:
        botName = 'Claudia';
        break;
    case 7:
        botName = 'Lydia';
```

```
        break;
    case 8:
        botName = 'Peter';
        break;
    case 9:
        botName = 'William';
        break;
}
```

Die Zufallszahl wird eine Zahl zwischen 0 und 9 sein, wir brauchen somit 10 Namen.

Event Draw:

```
draw_set_color(c_white);
draw_text(30, 30, 'Die Zahl lautet: ' + string(zufallsZahl));
draw_text(30, 50, 'Der Name des Bots lautet: ' + botName);
```

Zur Kontrolle geben wir auch die Zahl aus. So kannst Du selbst prüfen, ob alles stimmt. In einem Spiel würden wir uns das natürlich sparen.

Das war es schon. Um einfacher testen zu können, kannst Du noch eine Taste einbauen, mit der Du das Programm neu starten kannst.

Date Created

10. November 2016

Author

sven