Casino Würfel – Das Ein-Objekt-Spiel

Description

In diesem Tutorial machen wir ein kleines Spiel. Dabei geht es nicht um das Gamedesign, sondern mehr um die Technik. Wir entwickeln ein kleines Spiel, welches lediglich aus einem Objekt besteht. In der Windows-Version nutzen wir keine Skripte, einen Raum und nur wenige Grafiken. Diese wurden übrigens von Harald in Photoshop erstellt.

Wer gleich einen Blick darauf werfen will, kann es im Browser spielen. Das komplette Beispiel könnt ihr hier herunterladen. Der Code ist kommentiert, hier im Artikel werden aber die wichtigsten Stellen ausführlicher erklärt.

Das Spielprinzip

Ich wollte es einfach halten. Man ist in einem Casino und hat zwei Würfel. Es wird Geld gesetzt, der Spieler würfelt, die Bank würfelt und wer gewinnt, behält das Geld. Das geht so lange, bis man pleite ist. D. h. wir arbeiten mit Zufallszahlen und ein paar wenigen Variablen. Eine Besonderheit besteht darin, dass die Würfel nacheinander angezeigt werden. Das lösen wir über Alarme. Das Prinzip kennen manche vielleicht schon aus dem Memory-Tutorial.

Bei einem Unentschieden bleibt das Geld im Pott. Man kann es beim nächsten Wurf verlieren oder gewinnen.

Einziges Problem: Wenn man kein Geld mehr hat, aber es ist aufgrund eines Unentschiedens noch Geld im Pott, hat man es verloren. Aber die Regeln könnt ihr natürlich selbst bestimmen, wie ihr wollt.

Vorbereitung

Unseren Raum passen wir auf die richtige Auflösung, 1024×576 an. Ich habe diese Auflösung gewählt, um ein 16:9 Seitenverhältnis zu haben. Für das Beispiel wollte ich aber keine 1080p verwenden.

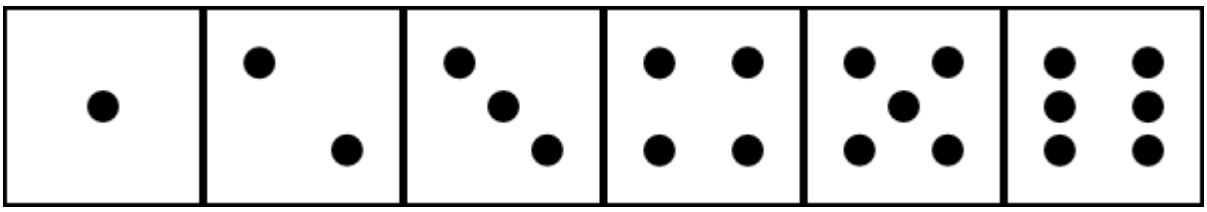
Grafiken

Als Sprites nehmen wir zwei Würfel, einen Hintergrund und ein Logo. Das Logo kann man, theoretisch, auch als Text erzeugen.

- **bg_table** ist unser Hintergrund
- **spr_dice1** ist der Würfel des Spielers
- **spr_dice2** ist der Würfel der Bank
- **spr_logo** ist unser Schriftzug

Das Hintergrundbild stellen wir im Raum ein.

Der Würfel ist einfach gehalten. Kann man auch direkt im GM erzeugen. Ich habe ihn mit Affinity Photo gemacht.



Der Würfel der Bank ist identisch, aber invertiert. Je nach Tisch und Farbverteilung bieten sich aber auch zwei bunte Würfel an.

Schrift

Ich verwende lediglich eine Schriftart, die ich **fnt_cassino** nenne. Für das Beispiel habe ich einen pixeligen Retro-Font genommen. Ihr seid natürlich frei, dies bei euch zu ändern.

Das eine Objekt

Zunächst stellt man sich die Frage, warum man überhaupt ein Spiel mit nur einem Objekt machen sollte. Ist es nicht besser, die Dinge ordentlich auszulagern? Mehrere Skripte, die ausgelagert mehr Übersicht bieten?

Im Prinzip ja. Und was ich gleich zeige, ist auch kein besonders guter Stil. Es soll aber einerseits die Möglichkeiten im GameMaker, aber auch gewisse Techniken der Programmierung zeigen. Gerade bei großen Projekten ist es irgendwann nicht mehr gut, wenn man zu viele Objekte und Skripte hat. Dann springt man von einem Fenster zum nächsten und hat Probleme, den eigenen Code zu verstehen. Und ja, manchmal macht es auch Spaß, einfach Code herunter zu tippen, wenn er am Ende funktioniert.

Im Beispiel habe ich das Objekt treffenderweise **obj_casino** genannt. Es beinhaltet 12 Events. Drei davon reagieren nur auf die Tastatur und sind nicht besonders spannend. Außerdem haben wir sechs Alarme. Schauen wir uns also die neun Events an.

Create-Event

```
randomise();    // Wir starten den Zufallsgenerator

// Variablen Würfel Spieler
pw1   = 0;
pw2   = 0;
psum  = 0;

// Variablen Würfel Cassino
cw1   = 0;
cw2   = 0;
csum  = 0;

canPlay  = true;   // Kann gespielt werden?
endResult = 0;   // Wer hat gewonnen? 0=kein Resultat; 1=Sieg; 2=Niederlage; 3

insertMoney = [5, 10, 25, 50, 75, 100]; // In diesen Schritten kann man sein G
playerMoney = 200;   // Startkapital
bank  = 0;   // Das gesetzte Geld bzw. das Geld, welches man gewinnen kann

gameOver = false;

stepSpeed = room_speed/2;  // Geschwindigkeit für den Fortschritt
halfW  = room_width/2;
halfH  = room_height/2;
```

Zunächst starten wir den Zufallsgenerator, bevor er vergessen wird. Wir brauchen drei Variablen für den Spieler. *pw1* und *pw2* sind die beiden Würfel, *psum* ist die Summe beider Würfel. Das Gleiche haben wir für das Casino. *insertMoney* ist ein Array. Der Spieler wird im Spiel in diesen sechs Schritten frei den Betrag wählen können, den er setzen möchte. Die restlichen Variablen sind selbstredend.

Step-Event

```
if (!gameOver) && (canPlay) && (mouse_check_button_released(mb_left))
{
 if (mouse_x>room_width-120) && (mouse_x<room_width-40)
 {
  var i = -1;
  if (mouse_y>80)  && (mouse_y<110) && (playerMoney >= insertMoney[0]) { i = 0
  if (mouse_y>140) && (mouse_y<170) && (playerMoney >= insertMoney[1]) { i = 1
  if (mouse_y>200) && (mouse_y<230) && (playerMoney >= insertMoney[2]) { i = 2
  if (mouse_y>260) && (mouse_y<290) && (playerMoney >= insertMoney[3]) { i = 3
  if (mouse_y>320) && (mouse_y<350) && (playerMoney >= insertMoney[4]) { i = 4
  if (mouse_y>380) && (mouse_y<410) && (playerMoney >= insertMoney[5]) { i = 5

  if (i > -1)
  {
   playerMoney -= insertMoney[i];
   bank += insertMoney[i];
   alarm[5] = 10;
  }
```

```
 }
}
```

Hier geht es darum, das Geld zu setzen. Gezeichnet wird das alles im **Draw-Event**, aber wir müssen ja irgendwo abfragen, was die Maus macht und ob sie an der richtigen Position ist.

Die Abfrage startet, wenn das Spiel noch nicht vorbei ist, der Spieler drücken darf und die linke Maustaste gedrückt wurde.

Dann fragen wir x- und y-Koordinate ab. Da sich alle Werte auf der selben x-Koordinate bzw. in einem Bereich befinden, müssen wir das nur einmal prüfen. Das ist unser *if (mouse_x>room_width-120) && (mouse_x<room_width-40)*

Die lokale Variable habe ich *i* genannt. Das soll auf die *for-Schleife* im Draw-Event Bezug nehmen. Nun prüfen wir, in welchem Bereich sich die Maus auf der y-Koordinate befindet, wenn die linke Maustaste gedrückt wurde. Entsprechend weisen wir *i* einen Wert zwischen 0 und 5 zu. Ist der Wert >-1, ziehen wir der Variable *playerMoney* Geld ab und geben es der Bank. Das schauen wir uns im Detail an.

```
playerMoney -= insertMoney[i];
bank += insertMoney[i];
```

insertMoney haben wir im Create Event definiert. Ein Array mit 6 Werten. Arrays beginnen bei der Zählung mit 0. Ist *i* bspw. 1, nehmen wir den zweiten Wert aus dem Array. Im Beispiel sind das 10$. Die werden dem Spieler abgezogen (Zeile 1) und der Bank gegeben (Zeile 2).

Die Abfrage muss so „kompliziert” sein, weil der Spieler „irgendwo” links klicken kann. Also auch zwischen den Werten.

Draw-Event

```
// Überschrift
draw_sprite_ext(spr_logo, 0, halfW+8, 48, 1, 1, 0, c_black, .5);
draw_sprite(spr_logo, 0, halfW, 40);

// Schrift
draw_set_font(fnt_cassino);
draw_set_valign(fa_middle);
draw_set_alpha(1);
draw_set_color(c_white);

// Geld des Spielers
draw_set_halign(fa_left);
draw_text(60, room_height-60, „Du hast: „ + string(playerMoney) + „$");
draw_set_halign(fa_right);
draw_text(room_width-60, room_height-60, „Einsatz: „ + string(bank) + „$");

// Geldeinsatz
draw_set_halign(fa_center);

if (!gameOver)
```

```
{
 for (var i=0; i<6; i++)
 {
  draw_set_color(c_white);

  if (playerMoney >= insertMoney[i])
  {
   if (mouse_x>room_width-120) && (mouse_x<room_width-40) && (canPlay)
   {
    if (i = 0) && (mouse_y>80) && (mouse_y<110) { draw_set_color(c_red); }
    if (i = 1) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 2) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 3) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 4) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 5) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
   }

   draw_text(room_width-80,100+60*i,string(insertMoney[i]) + „$");
  }
 }
} else {
 // Das Spiel ist vorbei!
 draw_set_color(c_black);
 draw_set_alpha(.5);
 draw_rectangle(halfW-212,halfH-92,halfW+228,halfH+108,0);
 draw_set_color(make_color_rgb(8,115,16));
 draw_set_alpha(1);
 draw_rectangle(halfW-220,halfH-100,halfW+220,halfH+100,0);
 draw_set_alpha(.5);
 draw_set_color(c_black);
 draw_text(halfW+4, halfH+4, „R = Neustart\nEsc = Ende");
 draw_set_alpha(1);
 draw_set_color(make_color_rgb(216,147,5));
 draw_text(halfW, halfH, „R = Neustart\nEsc = Ende");
}

draw_set_halign(fa_left);
draw_set_color(c_white);

if (!gameOver)
{
 // Würfel Spieler
 if (pw1 > 0)
 {
  draw_text(40, 180, „Deine Wurf:");
  draw_sprite(spr_dice1, pw1-1, halfW-120, 180);
 }

 if (pw2 > 0)
 {
  draw_sprite(spr_dice1, pw2-1, halfW+120, 180);
 }

 // Würfel Cassino
 if (cw1 > 0)
```

```
 {
  draw_text(40, 320, „Casino:");
  draw_sprite(spr_dice2, cw1-1, halfW-120, 320);
 }

 if (cw2 > 0)
 {
  draw_sprite(spr_dice2, cw2-1, halfW+120, 320);
 }
}

// Ergebnis verkünden!
draw_set_halign(fa_center);

if (endResult > 0)
{
 var endTXT = „";

 switch (endResult)
 {
  case 1:
   endTXT = „Du hast gewonnen!";
   draw_set_color(make_color_rgb(214,130,255));
  break;
  case 2:
   endTXT = „Du hast verloren!";
   draw_set_color(make_color_rgb(216,147,5));
  break;
  case 3:
   endTXT = „Es ist Unentschieden!";
   draw_set_color(make_color_rgb(245,246,121));
  break;
 }

 draw_text(halfW, 440, endTXT); // Das Resultat wird in der entsprechenden Far
}
```

Okay, das ist viel Code. Schauen wir es uns also in kleineren Blöcken an.

```
Der Anfang sollte klar sein. Wir zeigen das Logo an und machen das sogar mit e
```

```
// Geldeinsatz
draw_set_halign(fa_center);

if (!gameOver)
{
 for (var i=0; i<6; i++)
 {
  draw_set_color(c_white);

  if (playerMoney >= insertMoney[i])
```

```
  {
   if (mouse_x>room_width-120) && (mouse_x<room_width-40) && (canPlay)
   {
    if (i = 0) && (mouse_y>80) && (mouse_y<110) { draw_set_color(c_red); }
    if (i = 1) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 2) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 3) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 4) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
    if (i = 5) && (mouse_y>80+60*i) && (mouse_y<110+60*i) { draw_set_color(c_r
   }

   draw_text(room_width-80,100+60*i,string(insertMoney[i]) + „$");
  }
 }
} else {
 // Das Spiel ist vorbei!
 draw_set_color(c_black);
 draw_set_alpha(.5);
 draw_rectangle(halfW-212,halfH-92,halfW+228,halfH+108,0);
 draw_set_color(make_color_rgb(8,115,16));
 draw_set_alpha(1);
 draw_rectangle(halfW-220,halfH-100,halfW+220,halfH+100,0);
 draw_set_alpha(.5);
 draw_set_color(c_black);
 draw_text(halfW+4, halfH+4, „R = Neustart\nEsc = Ende");
 draw_set_alpha(1);
 draw_set_color(make_color_rgb(216,147,5));
 draw_text(halfW, halfH, „R = Neustart\nEsc = Ende");
}
```

Der größte Block ist der Geldeinsatz. Wird nur angezeigt, wenn das Spiel noch

Richten wir unsere Aufmerksamkeit auf die *for-Schleife*
. Hier wird das angezeigt, was wir im Step-Event bereits abfragen. Zugegebener

1. Wenn die Maus über dem richtigen Geldbetrag ist, wird es rot eingefärbt. A
2. Es werden nur die Werte angezeigt, die man tatsächlich ausgeben kann. Wenn

Letzteres bewirkt die Zeile *if (playerMoney >= insertMoney[i])*

Hier zeigen wir die Würfel an. Für Spieler und Casino gibt es je eine Textzeil

Wenn die Würfel gefallen sind, verkünden wir das Ergebnis. Dies machen wir einfach per Switch. Die entscheidende Variable ist *endResult*.

Alarme 0 bis 5

Damit das alles in so schönen Schritten abläuft, verwenden wir Alarme. Wir hab
Alarm 5 setzt alle Werte zurück. Dieser Alarm wird im *Step-Event*
aufgerufen. Von da aus geht es zu *Alarm 0* und dann der Reihe nach bis
Alarm 4, der Auswertung.

Alarm 5

```
endResult = 0;  // Resultat auf 0

// Variablen Würfel Spieler
pw1  = 0;
pw2  = 0;
psum  = 0;

// Variablen Würfel Cassino
cw1  = 0;
cw2  = 0;
csum  = 0;

canPlay  = false; // Man darf nichts mehr setzen
alarm[0] = stepSpeed/2; // Und wir starten!
```

Wir löschen das Resultat, setzen alle Werte auf 0, der Spieler kann nicht mehr
Alarm 0.

Alarm 0

```
pw1  = choose(1, 2, 3, 4, 5, 6);
alarm[1] = stepSpeed;
```

Das war es schon. Für den Würfel nehmen wir *choose()*. Man könnte auch
random_range()
verwenden, aber das ist kaum kürzer, da man runden, mit 100 multiplizieren un

Alarm 1

```
cw1   = choose(1, 2, 3, 4, 5, 6);
alarm[3] = stepSpeed;
```

Natürlich will auch das Casino würfeln.

Alarm 3

```
cw2   = choose(1, 2, 3, 4, 5, 6);
csum  = cw1+cw2;
alarm[4] = stepSpeed;
```

Und auch hier fällt der zweite Würfel und die Werte des Casinos werden addiert

Alarm 4

```
// Bei Gewinn wird ausbezahlt und der Einsatz auf 0 gesetzt.
// Bei einem Unentschieden bleibt das Geld drin.
// Bei einer Niederlage wird es gelöscht.
if (psum > csum)
{
 playerMoney += bank*2;
 bank = 0;
 endResult = 1;
} else if (psum < csum) {
 bank = 0;
 endResult = 2;
} else {
 endResult = 3;
}

canPlay = true;

if (playerMoney <= 0)
{
 gameOver = true;
}
```

Jetzt wird ausgewertet. Es gibt drei Zustände: gewonnen, verloren, unentschied

Im Prinzip war es das schon. Die drei Events, die ich hier unterschlage, sind **R**, **Esc** und **G**, die den Raum neu starten, beenden oder das *gameOver* auslösen, damit man es beim Testen leichter hat.

Date Created
29. Dezember 2021
Author
sven