

Das große Game Maker Tutorial

Teil I: Game Maker Grundlagen

von Sebastian Dürre

Inhalt:

1. Einführung
2. Der Umgang mit diesem Tutorial (lesen!)
3. Game Maker Grundlagen
 - Der Sprite Editor
 - Der Sound Editor
 - Der Hintergrund Editor
 - Schriftarten (Fonts)
 - Objekte
 - Räume
 - Das erste kleine Spiel

1. Einführung

Dieses Tutorial wurde von Sebastian Dürre geschrieben und darf nicht als eigenes Tutorial ausgegeben werden. Der Sprite und der Sound ist von mir, darf auch frei verwendet werden (ohne Credit-Eintrag o.ä.). Ich wünsche dir noch viel Spaß beim lesen dieses Tutorials und ich hoffe, dass es dir weiterhilft! ;-)

2. Der Umgang mit diesem Tutorial

Du hast dir den Game Maker von Mark Overmars (<http://www.gamemaker.nl>) heruntergeladen, weist aber nicht, wie du ihn benutzt und richtig anwendest? Dann ist dieses Tutorial genau das richtige für dich. Es erklärt alles von Grund auf, damit du auch einmal Spiele entwickelst. Wichtige Voraussetzungen von dir sind:

Die Freude am Spiele machen:

Du sollst immer Spaß am Spiele machen haben, denn das ist das wichtigste! Freude am Spiele machen bedeutet, dass du immer Spaß hast und nicht, weil du einen überzeugen willst, dass du gute Spiele machen kannst / wirst.

Nicht die Motivation verlieren:

Es gibt nicht viele, die Grafiken und Sounds machen können. Aber das erste Spiel muss so etwas auch nicht haben (allein der Inhalt zählt und Grafiken kann man immer machen!) Sei auch nicht verzweifelt, wenn eins deiner Skripte nicht funktioniert. Meist gibt es viele, die einfach vergessen, eine Klammer zu setzen oder eine Endlosschleife erstellen und damit das Spiel zum Absturz bringen. Das Problem löst sich und man bemerkt, dass es eigentlich ein ganz kleiner Fehler war!

Geduld:

Das ist der wichtigste Punkt, denn erwarte vom Game Maker nicht, dass du in kurzer Zeit ein Doom 3 Spiel entwickelst. Der Game Maker wurde zwar mit 3D Funktionen ausgestattet, aber trotzdem solltest du nicht mit 3D Anfangen. Der Game Maker wurde laut Mark auch nur für 2D Spiele entwickelt (und 2D macht immer Spaß).

Beachte auch, dass an den heutigen Spielen meist über 100 Leute sitzen und die dann trotzdem über 3 Jahre Entwicklungszeit brauchen. Du sitzt alleine (oder in einem Team von ungefähr 10 Leuten, dessen Aufgaben gezielt verteilt sind) an einem Projekt, dass in 3 bis 4 Monaten fertig ist.

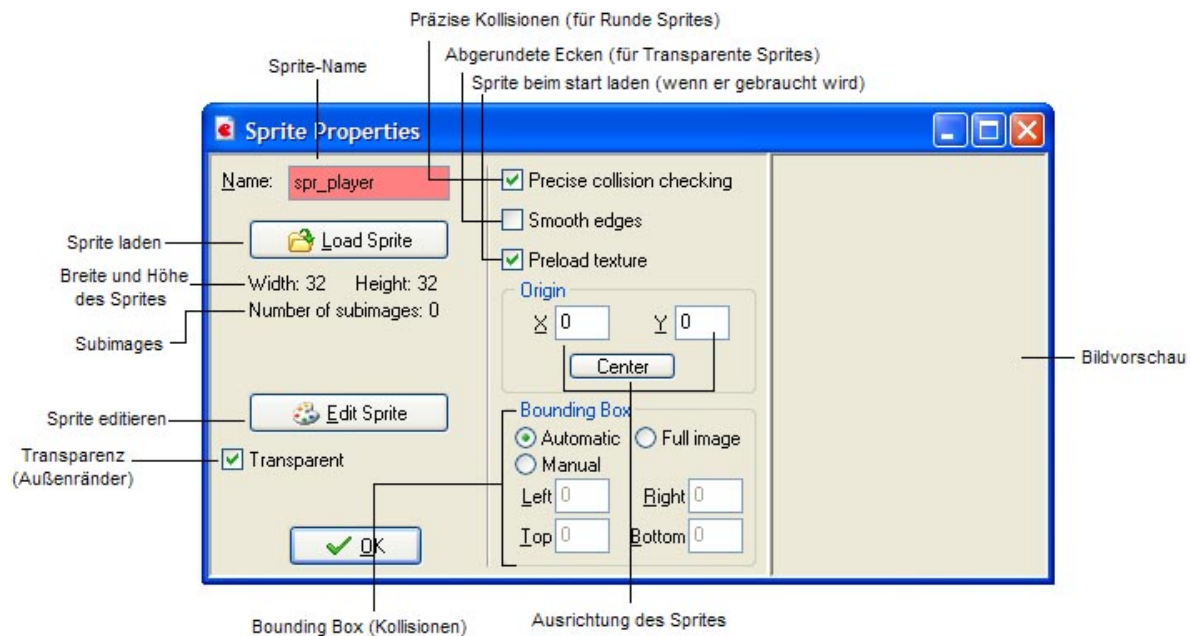
Das war jetzt auch genug geredet! Lass uns Anfangen! ;)

3. Game Maker Grundlagen

Hinweis: Wir lassen manche Editoren aus, die wir für unser Tutorial nicht brauchen, oder kommen später darauf zurück.

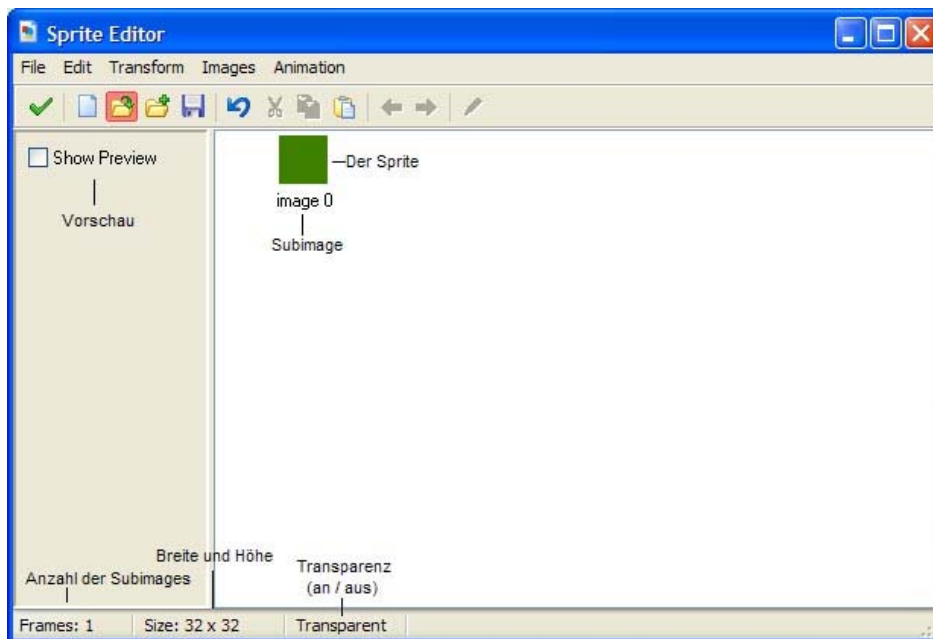
Der Sprite Editor

Der Game Maker wurde mit einem Internen Grafik-Editor ausgerüstet, damit du deine Bilder nicht in Paint oder in ähnlichen Programmen malen musst. Sprite heißt ins Deutsche wörtlich übersetzt: „Kobold“. Kobolde sind für uns die „Grafiken“ oder „Bilder“. Um einen Sprite hinzuzufügen, gehen wir zu dem Ordner „Sprites“ und klicken mit der rechten Maustaste. Es öffnet sich ein Menü und wir klicken auf „Add Sprite“.



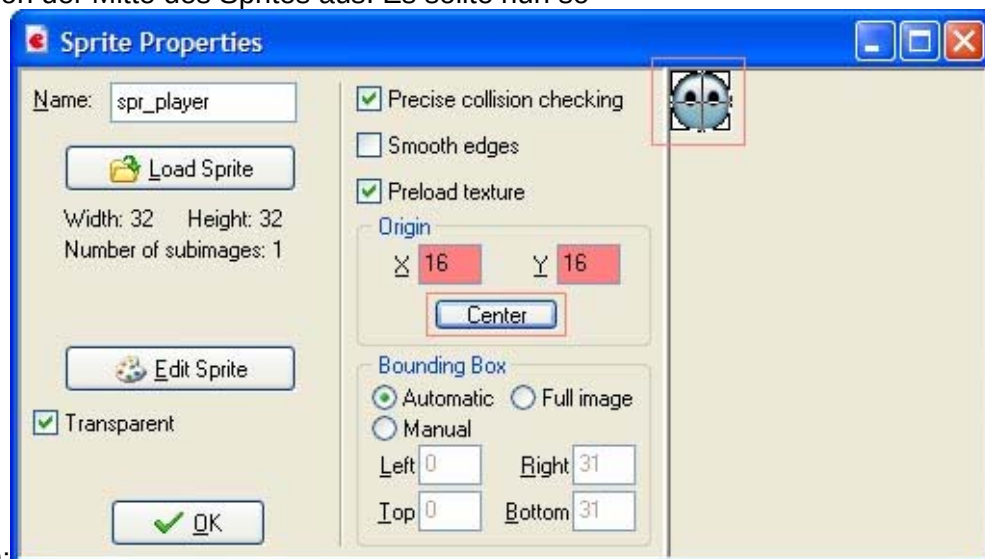
Es öffnet sich ein Fenster mit vielen Optionen. Als Name geben wir „spr_player“ ein. Dieser Sprite wird später unser Spieler. Manche schreiben auch „spritePlayer“ / „sprPlayer“ / „sprite_player“ / „player_spr“ oder „player_sprite“. Es ist egal, wie man es schreibt. Wie du aber gesehen hast, darf im Name kein Leerzeichen oder Sonderzeichen (+, !, ., ...) stehen. Ich habe schon immer „spr_player“ genommen, weil dies am übersichtlichsten ist. Die anderen Optionen lassen wir erst mal aus (darauf kommen wir später zurück). Wir gehen auf „Edit Sprite“.

Es öffnet sich noch ein Fenster. Links kannst du eine Trennlinie sehen. Dies trennt den Editor von der „Preview“ also „Vorschau“. Dies interessiert uns noch nicht. Wir kommen nach dem fertigem Bild darauf zurück. Auf der anderen Seite sehen wir ein dunkelgrünes Kästchen. Darunter steht „image 0“. Dort können wir sehen, welches „Subimage“ dies ist. Der Game Maker fängt hier immer bei 0 an (das wirst du an manchen Stellen des Game Makers noch öfters sehen). Wir machen einen Doppelklick auf das Kästchen. Wer mit dem Paint vertraut ist, kennt hier sicher schon einige Sachen. Mit einem Doppelklick lässt sich dann der Zeichen-Editor öffnen. Wir brauchen ihn im Moment nicht, da ich eigene Grafiken dabei habe ;). Unten siehst du ein Bild des Sprite Editors:



Klicke auf dem im Bild rot markierten Button. Nun gehst du in unser Tutorial-Verzeichnis und dann unter „Sprites“. Wähle dort die „spr_player“ Datei. Nun hat sich das Kästchen in unser Bild geändert. Wir gehen nun auf den grünen Haken. Nun öffnet wird unser altes Fenster sichtbar.

Klicke auf den „Center“ Button. Er ändert die Ausrichtung des Sprites in der Mitte d.h. alles passiert von der Mitte des Sprites aus. Es sollte nun so

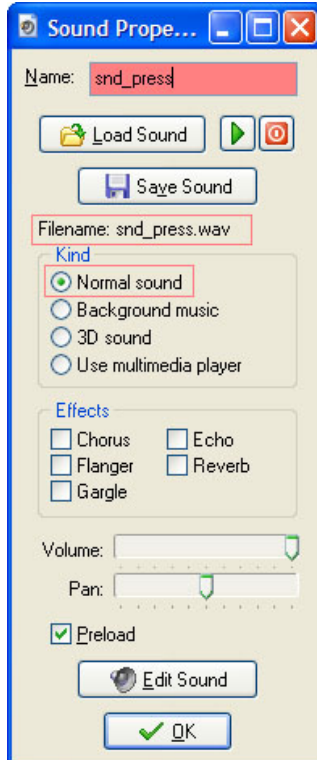


aussehen:

Mit dem Sprite Editor sind wir fertig. Du kannst (wenn du möchtest, dich noch mehr mit dem Sprite Editor vertraut machen, wenn du willst. Ich fahre normal weiter fort.

Der Sound Editor

Damit dein Spiel auch Geräuschvoll ist, brauchst du Sounds und Musik. Der Sound Editor ist sehr leicht zu bedienen und nicht so komplex, wie der Sprite Editor (kurz: er ist eigentlich ist der leichteste Editor des Game Makers).



Das ist der Editor. Als Namen geben „snd_press“ („snd“ = Sound) ein. Wir laden den Sound „snd_press“ aus dem Verzeichnis „Sounds“. Dort wo „Kind“ also „Typ“ steht, nehmen wir das Kontrollfeld „Normal Sound“, damit der Game Maker von Sound und Musik unterscheiden kann. Du kannst den Sound abspielen, indem du auf den Grünen Pfeil (rechts neben Load Sound) klickst. „Volume“ ist die Lautstärke unseres Sounds und „Pan“ die Position des Sounds (linker / rechter / beide Lautsprecher). Die Effekte waren für mich nie zu gebrauchen, du kannst aber ausprobieren, wenn du möchtest (Registrierte Version des Game Makers erforderlich). Wir können den Sound Editor jetzt schließen.

Der Hintergrund Editor

Wir werden den Hintergrund Editor nur kurz überfliegen, denn wir werden ihn für unser Tutorial nicht brauchen.

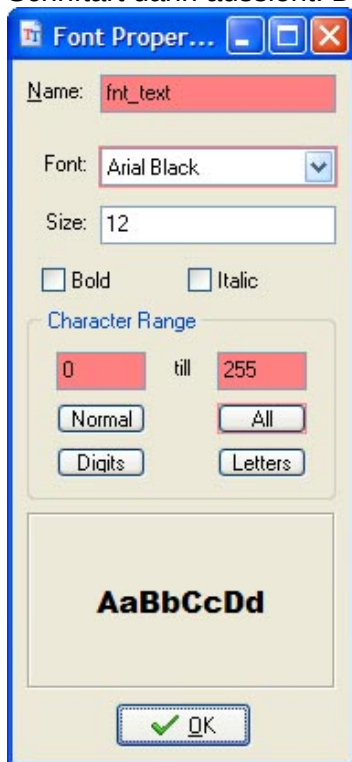
Der Hintergrund Editor ist dem Sprite Editor sehr ähnlich. Er ist dazu da, um dem Level einen Hintergrund zu geben. Die drei Kontrollfelder habe ich bereits im Sprite-Editor erklärt. Klicke auf „Load-Background“, um einen Hintergrund zu laden und „Edit Background“, um ihn zu Editieren. Du kannst ihn benutzen, aber für unser Tutorial ist er nicht wichtig.

Schriftarten (Fonts)

Schriftarten sind sehr wichtig für ein Spiel. Du möchtest einen Text darstellen lassen, aber in einer anderen Schriftart. Deswegen kannst du Schriftarten laden, die sich in deinem Windows-Ordner befinden. Wenn du später etwas mehr über die Codesprache des Game weisst, kann der Game Maker auch selber gemalte Fonts laden. Dies ist aber etwas schwerer und du brauchst auch eine gute Grafik. Das ist dafür aber das beste. Ist dem Sound-Editor in seinem aufbau ähnlich. Oben als Name geben wir „fnt_text“ („fnt“ = font) ein. Die Schriftart kannst du dir selber aussuchen (ich habe Arial-Black genommen ;).

Size ist die Größe des Textes. Ich habe sie auf 12 gelassen, denn größer muss der Text nicht sein Bold und Italic legen fest, ob der Text fett oder kursiv sein soll.

Die „Character Range“ legt die Zeichen fest. Standard beim Game Maker ist immer „from 32 till 127“ Standard. Leider werden da aber keine Üs / Äs oder Ös dargestellt. Um Das zu ändern, drücken wir den Button „All“. Nun kann der Game Maker alle Zeichen darstellen (sonst wäre dort immer ein Leerzeichen gewesen). Unten kannst du noch sehen, wie die Schriftart dann aussieht. Der Schriftarten-Editor sollte nun so aussehen:

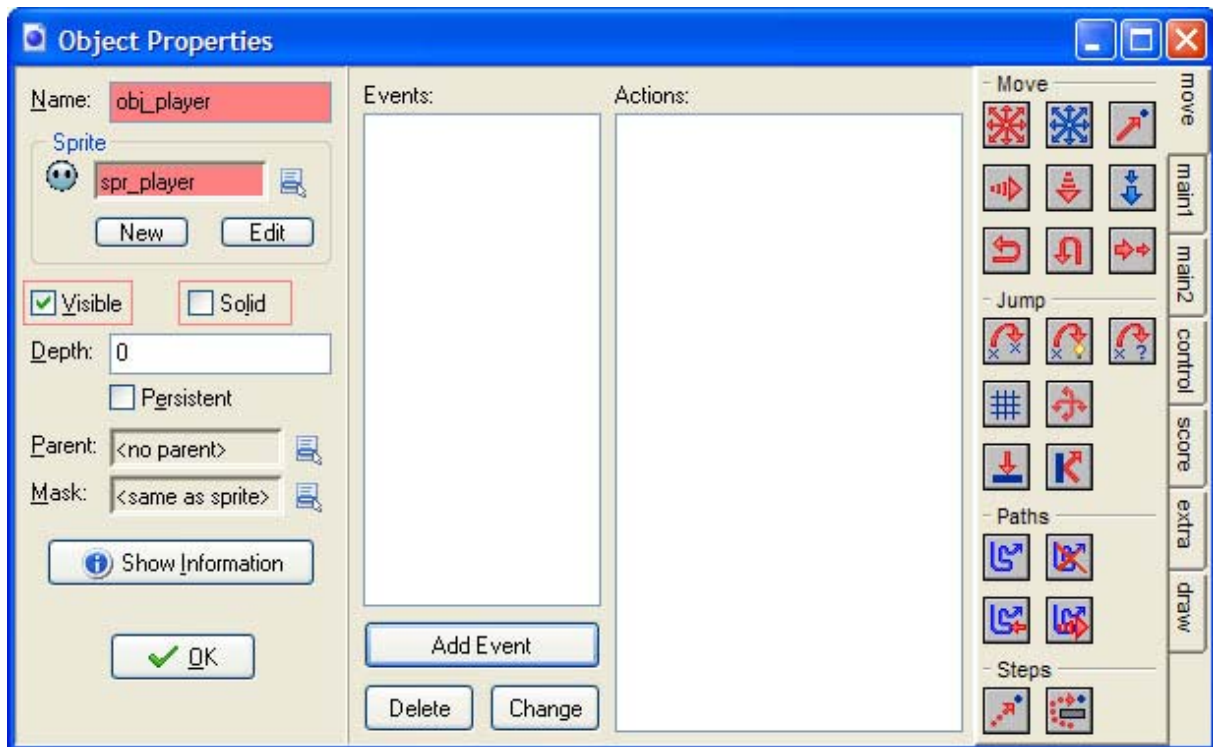


Wir haben jetzt auch unsere eigene Schriftart für unser Spiel und können ihn jetzt schließen. Zum Font-Editor ist zu sagen:

Wenn du dein Projekt in einer gm6 an einen anderen weiter gibts und eine heruntergeladene Font nimmst, ist die auf dem des Partners sicher nicht enthalten. Die musst du ihm dann selber schicken. Der Vorteil des Game Makers ist bei der Exe: Er importiert die Font gleich mit (d.h. auch eigene Schriftarten kann der Game Maker anzeigen; egal, ob ein Spieler diese Font besitzt, oder nicht).

Objekte

Dies ist eigentlich der wichtigste Teil eines Spiels. Objekte (kannst du dir eigentlich denken) sind die Objekte im Spiel. Sie führen etwas aus, fragen ab, zeichnen, rechnen, usw. Für Jedes Objekt gibt es den Objekt-Editor, den wir uns jetzt mal anschauen werden.



Ziemlich viel für den Anfang oder? Ja das ist es auch, denn Objekte haben auch das meiste an sich. Wichtig ist erst mal den Namen für das Objekt festzulegen. Ich habe es „obj_player“ genannt, weil dies auch unser Spieler ist. Nun müssen wir den Sprite für das Objekt festlegen. Klickt dazu auf das Menübild über den Edit Button. Nun wählt ihr unseren zuvor geladenen Sprite („spr_player“). Ich erkläre noch mal den „New“ und „Edit“ button. Der „New“-Button ist dazu da, um einen Neuen Sprite zu erstellen und ihn gleich danach zu definieren (bequemer). Ich mache es aber auf die längere Art. Der „Edit“ Button ist dazu da, um den Definierten Sprite zu Editieren. Dies kommt vor, wenn man keine Lust hat, den Sprite zu suchen. Ich benutze diesen Button öfters, weil er einfach mehr Zeit spart. Achtet jetzt darauf, dass die beiden Haken unter dem „New“- und „Edit“ Button so gesetzt sind, wie im Screen (so sind sie Standard immer gesetzt). Depth lassen wir erst mal so. Dies ist die Tiefe des Objektes. Z.B. möchte man manche Objekte im Hintergrund haben und manche sollen sich im Vordergrund aufhalten. Beachten ist aber: Je niedriger, desto weiter vorne und je höher, desto weiter hinten. D.h. ein Objekt mit Tiefe „-100“ würde vorne sein und „100“ dahinter. Den Haken Persistent lassen wir auch so. Dieser ist dazu da, damit das Objekt „Persistent“ wird, d.h. dass alles, was im Objekt verändert wird, gespeichert wird und ins nächste Level übernommen (auch die Position etc.). Da wir in unserem ersten Spiel nur ein Level haben, brauchen wir diesen Haken nicht zu setzen. Auf die beiden unteren Felder möchte ich nicht genau eingehen, weil wir diese noch nicht brauchen. „Parent“ ist dazu da, um dieses Objekt einem „Parent“, also „Elternobjekt“ zuzuordnen. Z.B. wenn wir für eine Gruppe etwas machen wollen, erstellen wir ein „Parent-Objekt“ und Ordnen die Objekte dem „Parent-Objekt“ zu. Dies spart auch Geschwindigkeit. Mit dem Button „Show-Information“ könnt ihr die Informationen des Objektes abrufen. Es wird hier noch wenig stehen, weil wir noch keine Aktionen und Events erstellt haben.

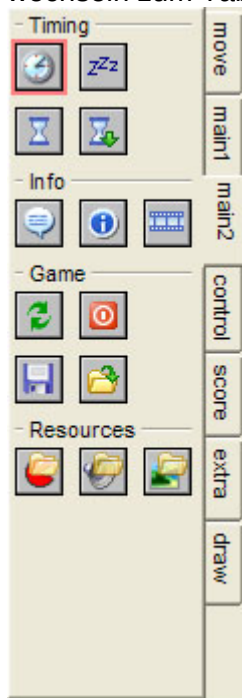
Objekte – Events

Achtung, denn dies wird der längste Teil von der Game Maker Grundlagen!

Damit unser Objekt auch etwas machen kann, müssen wir Events erstellen. Es gibt sehr viele Events für verschiedene Aktionen. Wir klicken auf den Button „Add-Event“. Es öffnet sich der „Event-Selector“:

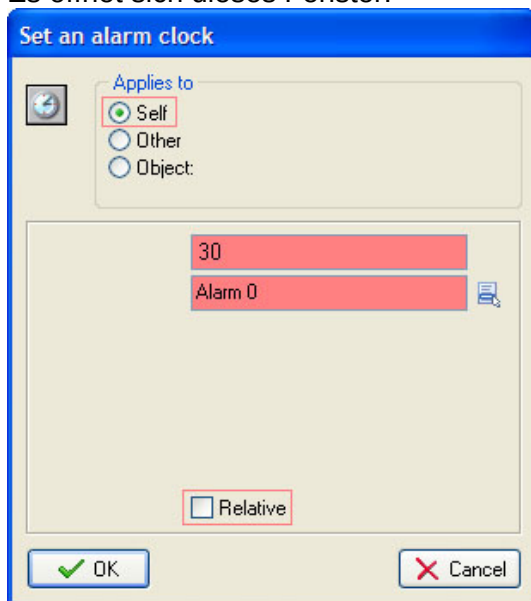


Wir wählen nun das markierte „Create“-Event. Das Create Event wird ausgeführt, sobald das Objekt erstellt wird. Nun erscheint ein Create-Bereich. Jeder Game Maker Benutzer fängt mit D&D (Drag & Drop) an, um den Game Maker kennen zu lernen und bildet sich später auf GML (Game Maker Language) fort. Nun haben wir links einen Tab, der „Move“ heißt. Wir wechseln zum Tab, das main2 heißt. In ihm sollte sich das Befinden:

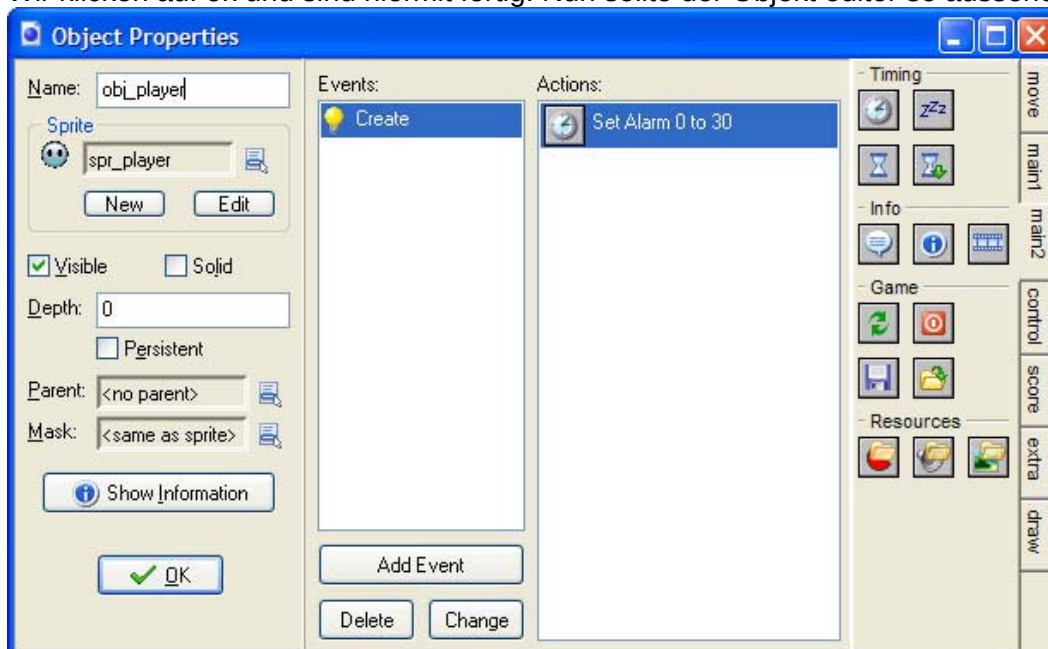


Klicken wir nun auf den Markierten Button mit der Uhr. Wir halten gedrückt und ziehen ihn auf die gegenüberliegende Seite, wo unser „Create“ steht.

Es öffnet sich dieses Fenster:

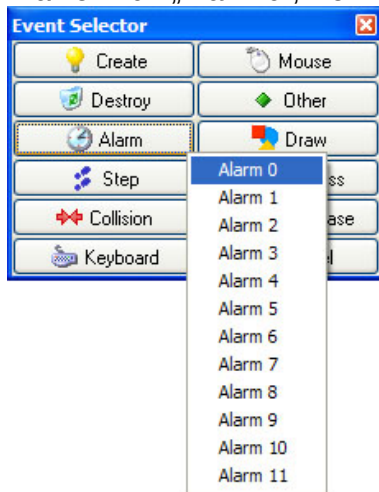


Diese Aktion ist dazu da, um etwas nach einer bestimmten Zeit zu machen („Alarm“). Wir können nun wählen, ob diese Aktion nur für dieses Objekt („Self“), für das andere Objekt, mit dem etwas passiert (z.B. bei einer Kollision) („other“), oder für ein ganz bestimmtes Objekt, was man wählen kann („Object“). Im ersten Feld wählen wir 30. Ein Spiel läuft standard mit 30 FPS. Weil wir nun 1 Sekunde haben möchten, müssen wir hier 30 (= eine Sekunde). Im Feld darunter geben wir „Alarm 0“ ein. Wir haben insgesamt 11 Alarm-Events. Hier definieren wir den Alarm. Im Alarm-Event, lassen wir dann etwas geschehen (und dazu kommen wir auch gleich). Den „Relative“-Haken lassen wir unausgewählt. Relative heißt, dass der GM den Alarm hinzufügt, z.B. wenn man ihn verändern will. Dies ist beim Alarm aber ziemlich dumm (und ich selbst habe es noch nie gebraucht [auch nicht in GML]). Wir klicken auf OK und sind hiermit fertig. Nun sollte der Objekt editor so aussehen.

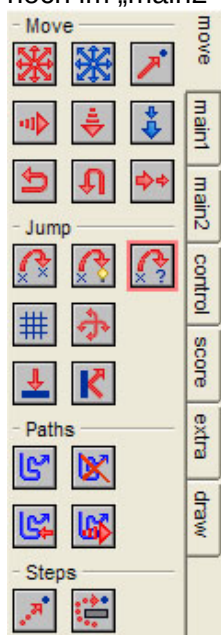


Wenn ihr mit eurer Maus über die Aktion fahrt, öffnet sich eine Quick-Info und dort könnt ihr die Infos noch mal abrufen.

Jetzt möchten wir natürlich noch etwas nach 1 Sekunde geschehen lassen. Wir erstellen nun das „Alarm-Event“. Hier ist eigentlich nicht viel zu sagen. Wir Klicken wieder auf „Add Event“ und in unserem Event-Selector wählen wir nun „Alarm“. Es öffnet sich ein neues Menü. Wir Wählen nun „Alarm 0“, weil wir „Alarm 0“ ja auch definiert haben.



Jetzt haben wir nicht nur „Create“ sondern auch noch „Alarm0“, wo wir schön hin- und herschalten können. In unserem Spiel geht es darum, dass man ein Objekt anklicken muss, bevor es die Position wechselt und man dafür Punkte kriegt. Also möchten wir nun, dass das Objekt nach einer Sekunde die Position wechselt. Also machen wir das auch. Wir sind immer noch im „main2“-Tab und wechseln nun zum „move“-Tab.

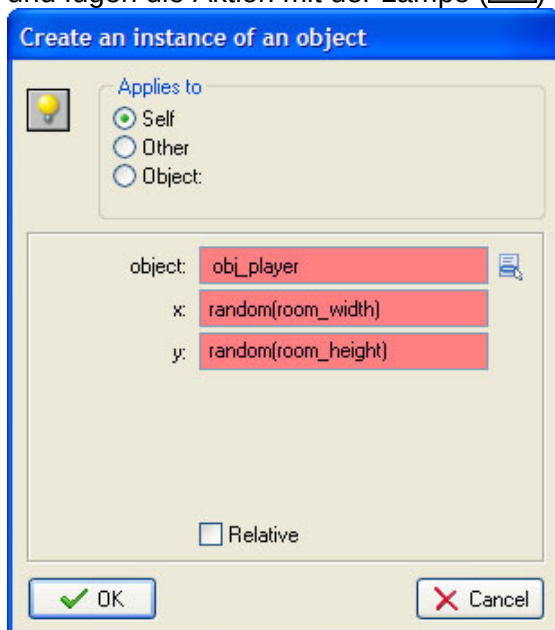


Nun wählen wir die rot-markierte Aktion. Diese nennt sich „Jump to a **random** position“. Das heißt der Computer wählt sich die Position selber aus. Wir ziehen die Aktion genau so wie auch davor wieder in das Feld. Es öffnet sich ein Fenster. Ich habe von diesem keinen Screenshot gemacht, weil wir diese beiden Felder einfach mit 32 Füllen. D.h. der Spieler wird an einem Raster von 32 Pixeln (Sprite-Größe und Sprite-Breite) geordnet. Unser Spiel würde jetzt schon laufen, aber mit den Rooms kennen wir uns noch nicht aus, also machen wir normal weiter. Jetzt würde in unserem Spiel der Spieler einmal in einer Sekunde zu einer unbekannten Position springen. Aber wir möchten, doch, dass er das öfters macht. Also kopieren wir die Aktion aus dem Create-Event in unser Alarm0-Event (unter die „Jump to a random position“-Aktion. Jetzt wird der Spieler es kontinuierlich machen. Jetzt kommt es zum 2. Teil des Objektes. Wir möchten, dass wenn man den Spieler mit der linken Maustaste erfasst hat, dass der Spieler dann weg von der Maus ist und dass man Punkte kriegt.

Um etwas mit der Maus zu machen, müssen wir ein „Mouse-Event“ hinzufügen. Wir öffnen wieder unseren Event-Selector und wählen das „Mouse-Event“. Jetzt kommt uns ein Fenster mit riesiger Auswahl entgegen. Ich erkläre kurz die verschiedenen Möglichkeiten:

- Left-, Right-, Middlebutton – Wird die linke, rechte oder mittlere Maustaste dauerhaft auf dem Objekt gedrückt (gehalten)?
- No Button – Wird keine Maustaste gedrückt?
- Left, Right, Middlepressed – Wird gerade die linke, rechte oder mittlere Maustaste auf dem Objekt gedrückt?
- Left, Right, Middlereleased – Wird die linke, rechte oder mittlere Maustaste auf dem Objekt losgelassen?
- Mouse Enter – Befindet sich die Maus gerade auf unserem Objekt?
- Mouse Leave – Befindet sich die Maus nicht auf unserem Objekt?
- Mouse wheel up – Wird das Mausexplorer nach oben gerollt?
- Mouse wheel down – Wird das Mausexplorer nach unten gerollt?
- Global Left-, Right-, Middlebutton – Wird die linke, rechte oder mittlere Maustaste dauerhaft gedrückt (gehalten)?
- Left, Right, Middlepressed – Wird gerade die linke, rechte oder mittlere Maustaste gedrückt?
- Left, Right, Middlereleased – Wird die linke, rechte oder mittlere Maustaste losgelassen?

Es gibt noch Joystick aber dies ist komplizierter und wer spielt GM Spiele mit Joysticks ? ;) Wir wählen jetzt **Left Pressed**. Dies ist unser letztes Event. Wir gehen jetzt zum „main1“-Tab und fügen die Aktion mit der Lampe (💡) ein.



Bei „object“ wählen wir unseren „obj_player“ aus.

Jetzt kommt ein Teil, der sich schon leicht an GML heranhängt. Eigentlich ist es schon GML nur in diesem Fenster. Bei „x“ geben wir *random(room_width)* ein. Random bedeutet soviel wie „Unbekannt“, was uns bei „Jump to a Random position“ schon aufgefallen ist.

„room_width“ heißt „Raum-Länge“, denn wir möchten ja nicht, dass der Spieler außerhalb des Raumes springt. Bei „y“ geben wir *random(room_height)* ein. Es ist genau das gleiche wie gerade eben nur dass mit room_height“ die Größe des Raumes gemeint ist. Den Relative haken lassen wir so (der ist bei einer Angabe mit *random* eigentlich auch Sinnlos).

Übrigens der Relative Haken ist dazu da, um etwas herauf oder herunterzuzählen. Man kann ihn auch benutzen, damit etwas von dem Objekt stattfindet. D.h. wenn man mit aktiviertem Relative-Haken etwas eingibt (z.B. 15), passiert es 15 Pixel vom Spieler entfernt.

Damit nach dem Mausklick nicht noch der alte übrig bleibt, müssen wir diesen jetzt zerstören. Dies machen wir ganz einfach, indem wir die Aktion „Destroy the instance“

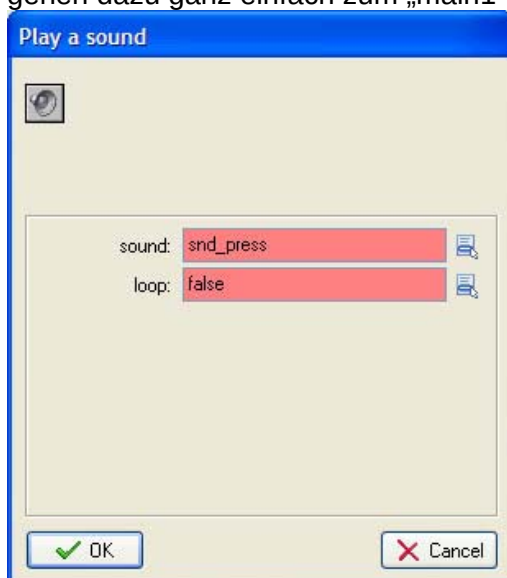
benutzen (). Diese zerstört die Instanz.

Wichtig: Instanz nicht gleich Objekt! Instanz ist das Objekt im Spiel und das Objekt im Editor. Wenn wir ein Objekt zerstören möchten, also aus der Ressource, müssten wir das mit GML machen und so was wird sowieso fast nie gebraucht. Zum Schluss möchten wir nur noch, dass wir einen Punkt kriegen, wenn wir getroffen haben. Wir gehen nun zum „score“-

Tab. Wir wählen nun „Set the Score“ (). Es öffnet sich dieses Fenster:



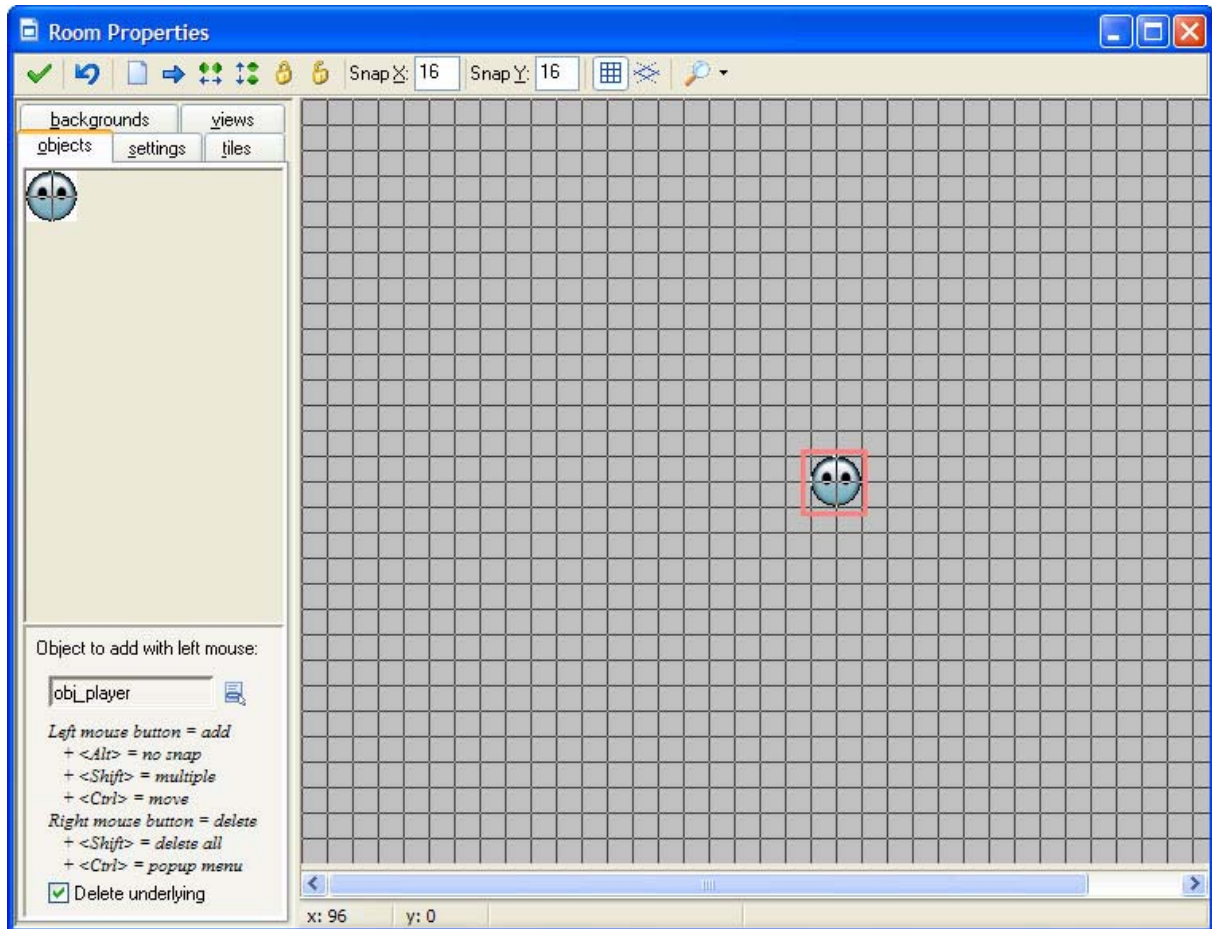
In das Feld „new score“ geben wir eine eins an. Dumm ist, dass wir immer nur einen Punkt haben und nicht draufgezählt bekommen. Deshalb aktivieren wir jetzt unseren kenn gelernten „Relative“-Haken. Jetzt kriegen wir immer einen Punkt draufgezählt, wenn wir den Spieler berühren. Jetzt möchten wir natürlich noch, dass unser geladener Sound abgespielt wird. Wir gehen dazu ganz einfach zum „main1“-tab und fügen die Aktion „Play a Sound“ hinzu (.



In das obere Feld kommt unser Sound hinein, indem wir auf das Icon klicken und unseren Sound auswählen. „loop“ müssen wir so lassen, denn „loop“ heißt soviel wie „wiederholen“ und wir möchten ja, dass der Sound nur einmal abgespielt wird. Jetzt ist unser Objekt fertig!

Räume

Damit unser Spiel einen Raum bzw. ein Level bekommt, fügen wir einen Raum hinzu. Nun kommt uns dieses Fenster zum Vorschein:



Wir platzieren unseren Player einfach in der Mitte des Rooms, indem wir auf die leere Fläche, wo unser Player abgebildet ist, raufklicken und dann „obj_player“ nehmen.. Wir können unser Spiel mit F5 (oder oben im Menü den grünen Pfeil drücken) das Spiel starten. Es sollte jetzt alles so laufen, wie wir es gewollt haben.