



Eigene Levels für SpinOff erstellen

# Level Construction Kit

Stand: 26. Februar 2002

Dateiname(Version): SpinOffLevelConstruct\_2.doc

Die aktuelle Version findet sich unter <http://spinoff.4fo.de>

Autor:

Martin Simeth

E-Mail: [MSimeth@gmx.de](mailto:MSimeth@gmx.de)

Download:

Web: [spinoff.4fo.de](http://spinoff.4fo.de) oder [www.4fo.de/download](http://www.4fo.de/download)

<b>1</b>	<b>EINLEITUNG .....</b>	<b>4</b>
<b>2</b>	<b>SPINOFF LEVELDESIGN .....</b>	<b>4</b>
<b>3</b>	<b>MEIN ERSTES LEVEL .....</b>	<b>5</b>
<b>4</b>	<b>MIT DEM EDITOR DEN LEVEL ERWEITERN.....</b>	<b>6</b>
<b>5</b>	<b>DIE SPIELELEMENTE .....</b>	<b>9</b>
5.1	Akteure .....	10
5.2	Effekte.....	10
5.3	Magie.....	11
5.4	Objekte .....	11
<b>6</b>	<b>FORTGESCHRITTENES LEVELDESIGN .....</b>	<b>12</b>
6.1	Manipulation von Faces .....	12
6.2	Manipulation von Brushes.....	13
6.3	Modelle .....	13
6.4	Beleuchtung.....	15
6.5	Neue Texturen.....	16
6.6	Ein neuer Himmel.....	17
6.7	Compiler Optionen .....	18
6.8	Der Levelpreviewer .....	18
<b>7</b>	<b>TIPPS .....</b>	<b>19</b>
<b>8</b>	<b>ANHANG ENTITY EIGENSCHAFTEN.....</b>	<b>19</b>
8.1	PlayerStart .....	20
8.2	ChangeLevel.....	20
8.3	Artefact .....	21
8.4	Marble.....	21
8.5	Music.....	21

<b>8.6</b>	<b>Design</b> .....	<b>22</b>
<b>8.7</b>	<b>Resize</b> .....	<b>22</b>
<b>8.8</b>	<b>Door</b> .....	<b>22</b>
<b>8.9</b>	<b>InstantDeath</b> .....	<b>23</b>
<b>8.10</b>	<b>Effects</b> .....	<b>23</b>
<b>8.11</b>	<b>AnimatedTexture</b> .....	<b>23</b>
<b>8.12</b>	<b>ChaosTexture</b> .....	<b>23</b>
<b>8.13</b>	<b>Corona</b> .....	<b>24</b>
<b>8.14</b>	<b>BeamField</b> .....	<b>24</b>
<b>8.15</b>	<b>ElectricBolt</b> .....	<b>24</b>
<b>8.16</b>	<b>ElectricBoltTerminus</b> .....	<b>24</b>
<b>8.17</b>	<b>ForceField</b> .....	<b>25</b>
<b>8.18</b>	<b>Light</b> .....	<b>25</b>
<b>8.19</b>	<b>Spotlight</b> .....	<b>25</b>
<b>8.20</b>	<b>Sunlight</b> .....	<b>25</b>
<b>8.21</b>	<b>DynamicLight</b> .....	<b>26</b>
<b>8.22</b>	<b>Camera</b> .....	<b>26</b>
<b>9</b>	<b>ANHANG BEGRIFFE</b> .....	<b>27</b>

# 1 Einleitung

SpinOff ist das ideale Spiel, um sich selbst einmal als Spieleentwickler zu betätigen. Sowohl als Anfänger, wie auch als Semiprofi: Den Möglichkeiten sind keine Grenzen gesetzt. Um möglichst früh mit den Begrifflichkeiten der Softwaretools vertraut zu werden, werden sie gleich zu Beginn eingeführt. Sie stehen in Klammern und sind fett geschrieben.

## 2 SpinOff Leveldesign

Ein SpinOff Level erstellt man in 4 Schritten:

1. Ein Level im Editor (**SpinOffEdit.exe**) zusammenstellen, erweitern korrigieren.
2. Den Level im Editor kompilieren.
3. Das Level im Levelpreviewer (**Preview.exe**) anschauen und Testspielen und notwendige Korrekturen in Schritt 1 einbringen.
4. Den Level an 4fo schicken, damit sie es auf der Game-Feature-Plattform zum Download anbieten.

Jeder Level besteht dabei aus 5 Grundelementen:

- Einem Himmel (**Sky**). Im Gegensatz zum echten Himmel, handelt es sich hierbei um einen geschlossenen (würfelförmigen) Raum, der ALLES umgibt. Da der Sky ein Würfel ist, der Himmel aber einen sphärischen Eindruck vermitteln soll, müssen die Texturen, die du für den Himmel verwenden willst entsprechend konstruiert sein. Achte immer darauf, daß sich nichts außerhalb dieses Himmels-Würfels befindet. Im StartLevel ist bereits ein Sky vorhanden, den du bei Bedarf vergrößern kannst.
- Objekte (**Brushes**) wie Würfel, Zylinder, Treppen, Bögen, welche die von der Kugel berührbaren Räumlichkeiten definieren. Mit ihnen baust du den Boden, Wände oder Schikanen.
- Subtraktive Objekte (**Cut Brushes**) mit ihnen kannst du z.B. Löcher in den Boden schneiden. Aber Achtung: Zuviel solltest du nicht mit ihnen arbeiten, sonst gibt es unschöne Seiteneffekte.
- Bewegliche Objekte (**Models**) mit ihnen realisierst du Türen oder Plattformen wie im Höllenlevel, eben alles, was sich an Geometrie im Spiel bewegen soll.
- Aktive Elemente (**Entities**). Mit ihnen realisierst du Kugeln, Sammelsteine, alle grafischen Effekte wie Feuer oder die blinkenden Pfeile (Übungslevel), alle Sorten von Magie, Beleuchtungseffekte, aber auch Sounds und die Musik.

Alle Brushes bestehen auch den sie umgebenden Flächen (**Faces**). Um Objekte richtig gut aussehen zu lassen, müssen sie eine Textur erhalten. Das muß du dir so vorstellen, als ob du auf alle Flächen eine Mustertapete (die Textur) klebst. Das trifft übrigens auch für die Cut Brushes zu. Die zu beklebende Fläche eines Cut Brushes ergibt sich aus den „Schnittflächen“ des ehemals unbeschädigten Objektes.

Die aktiven Elemente für dieses Spiel sind als vordefinierte Spielelemente verfügbar. Nachdem du ein Level konstruiert hast, mußst du es im Editor kompilieren: Du erhältst ein \*.bsp File. Dieses ist der später einmal von SpinOff spielbare Level. Ich sage später einmal, da der Level von 4fo ja noch freigegeben werden muß. Es läßt sich also noch nicht direkt mit

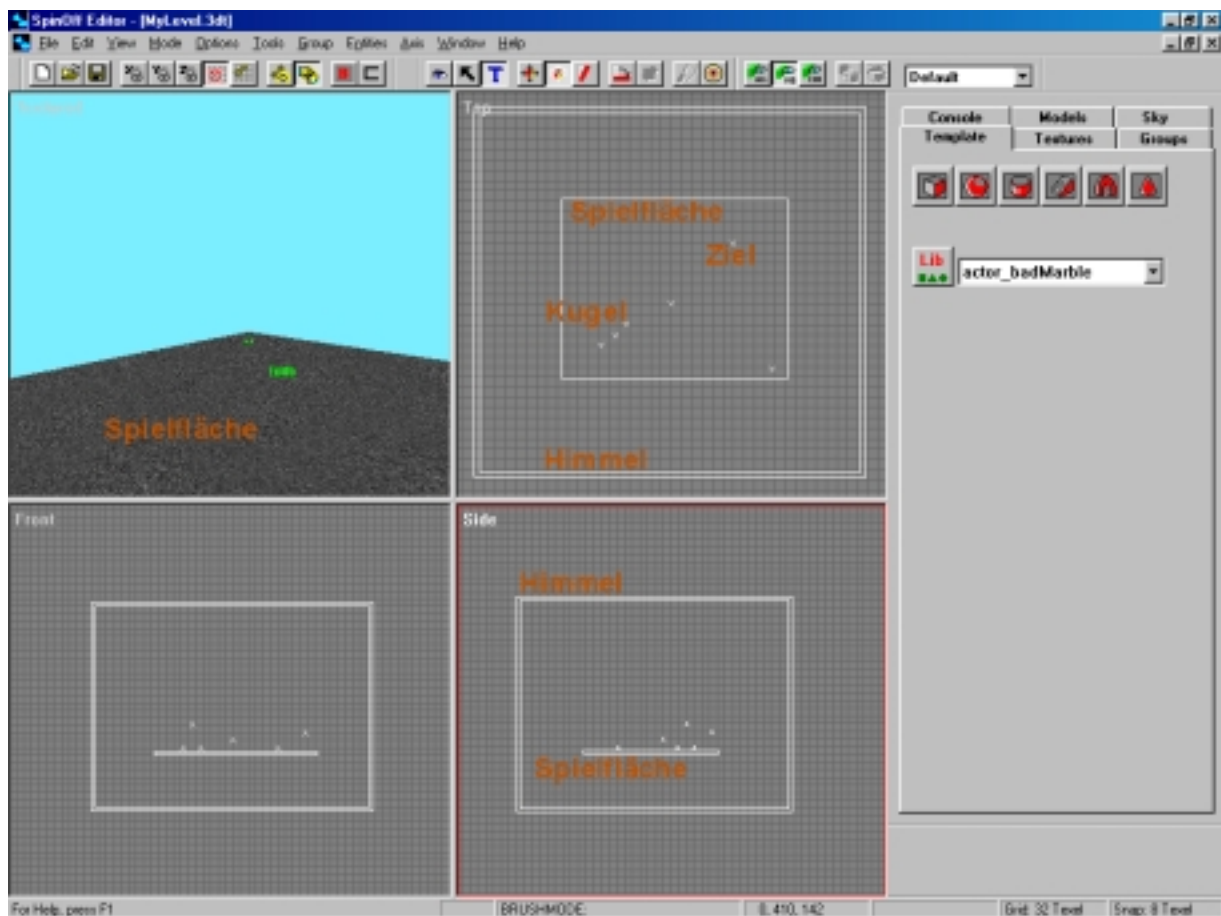
SpinOff spielen. Um deine Ungeduld bis dahin nicht allzusehr zu strapazieren, habe ich einen Levelpreviewer beigelegt. Der zeigt dir, wie dein Level aussieht, und wie es sich „anfühlt“.

### 3 Mein erstes Level

Genug der Theorie: Du baust jetzt das erste eigene Level!

Dazu startest du den SpinOff Editor **SpinOffEdit**, und lädst das im art-Verzeichnis befindliche Startlevel **MyLevel.3dt**. Um einem versehentlichen Überschreiben vorzubeugen, ist diese Datei schreibgeschützt. Wenn du also später einmal eigene Levels abspeichern willst, dann bitte unter einem anderen Namen.

Der Bildschirm des Editors besteht aus vier Hauptbereichen: das **Menu** oben, die **Werkzeugleiste** darunter, die vier **Editierfenster** und die sechs **Register** rechts.



Von den vier Editierfenstern zeigt das linke obere Fenster eine 3D-Ansicht des Levels (den Sky allerdings ohne Textur) aus einer beliebig wählbaren Kameraperspektive. Die drei anderen Fenster zeigen die Aufsicht (rechts oben) und zwei Seitenansichten (unten) des Levels.


Zum Navigieren in den 2D-Editierfenstern hält man die Leertaste gedrückt, und kann mittels der Maus den Fensterinhalt verschieben. Die einzelnen Fenster aktiviert man, indem man sie mit der Maus anklickt. Die 2D-Fenster können durch die '+' und '-' Taste gezoomt werden.

Um dich im 3D-Fenster zu bewegen, klickst du  in der Werkzeugleiste an.

Deine Betrachtungsposition kannst du jetzt durch Benutzung der Maus verändern.

- Für Bewegungen in der Horizontalen klickt man die linke Maustaste.
- Drehungen im Raum bewirkt man durch drücken der rechten Maustaste.
- Drückt man beide Maustasten gleichzeitig, kann man sich in der Szene rechts/links, bzw. hoch/runter bewegen.


Im Startlevel sind bereits alle wesentlichen Elemente, die du brauchst, vorhanden: Ein Himmel, eine Spielfläche, die Spielkugel (**Marble**), ein Sammelstein, ein Ziel (**Levelchange**), sowie etwas Musik.


Um das Startlevel gleich einmal auszutesten, kompilierst du es, indem du auf  und danach im auftauchenden Dialog auf OK drückst. Nach fertiger Kompilation startet der Previewer automatisch, oder du startest die Applikation **preview.exe** im bin Verzeichnis. Voila, das erste Level.


Du hast jetzt alle Schritte des Leveldesigns im Schnelldurchgang durchgeführt. Um dem Level ein paar persönliche Noten hinzuzufügen, musst du dich allerdings etwas intensiver mit dem Editor auseinandersetzen.

## 4 Mit dem Editor den Level erweitern

Der Editor wird in drei Modi betrieben:

View: durch anklicken von  wechselt man in den View mode. Dabei kann man die Kameraposition im 3D-Fenster verschieben, wie du es im vorigen Kapitel schon gelernt hast.


Edit: durch drücken von  kann man bestehende Objekte editieren.


Template: durch auswählen von  können neue Objekte in das Level eingebaut werden.

Im rechten Teil des Editors befinden sich 6 Register:

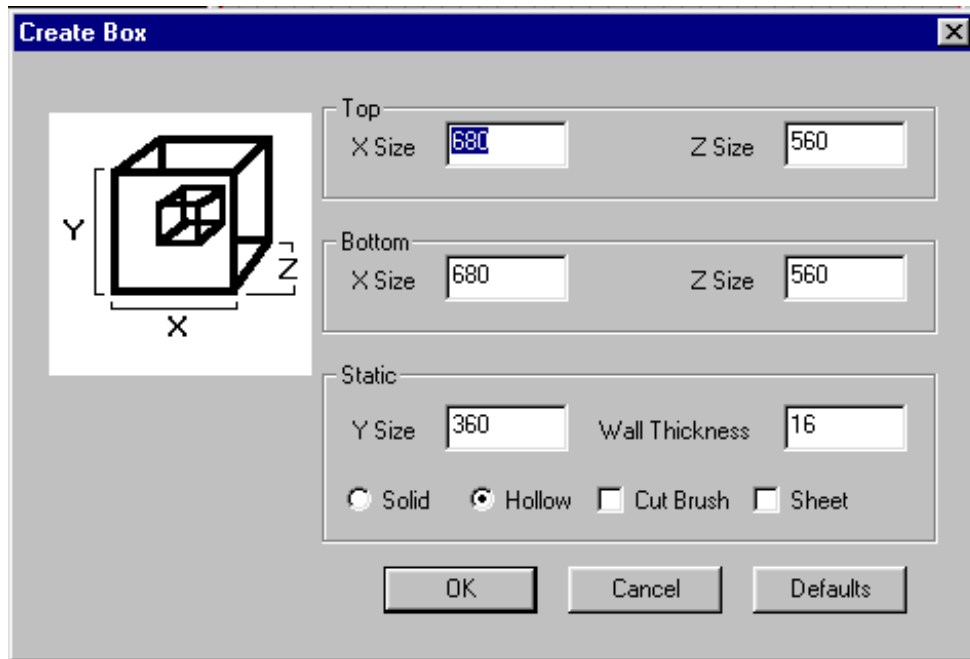
- Template:** Hier gibt es vordefinierte Brushes und die Spielelemente (**Lib**) für das Leveldesign.
- Textures:** Hier sind die Texturen zum „bekleben“ der Faces: Schau dir mal die Vorhandenen an.
- Groups:** Dient der Gruppenbildung. Alle vordefinierten Spielelemente sind bereits als Gruppe definiert, so kannst du sie einfach selektieren und verschieben.
- Console:** Hier gibt es Nachrichten vom Editor und dem Compiler.
- Models:** Die beweglichen Objekte definierst du hier. Und zwar sowohl ihre Zusammensetzung, als auch ihre Bewegungen.
- Sky:** Hier definierst du die Texturen für die 6 Seitenflächen des Himmels.

Du willst jetzt das Startlevel ein wenig ausbauen, indem du ein Loch in den Fußboden schneidest?

Du klickst also auf das  (Neue Objekte generieren), und wählst das Register **Template**.

Du willst das Loch quadratisch machen, und wählst den Würfel .

Der Create Box Dialog erscheint:





Sehr wichtig sind die vier unteren Einstellungen:

Einen soliden Würfel erhältst du, wenn du **Solid** wählst. Einen hohlen Würfel (also z.B. ein Raum ohne Türen und Fenster) ergibt sich, wenn du **Hollow** auswählst. Der Würfel wird „subtraktiv“, wenn du **Cut Brush** anklickst. Dann schneidet er ein quadratisches Loch in den Level.

Nicht näher behandeln will ich vorerst **Sheet**.

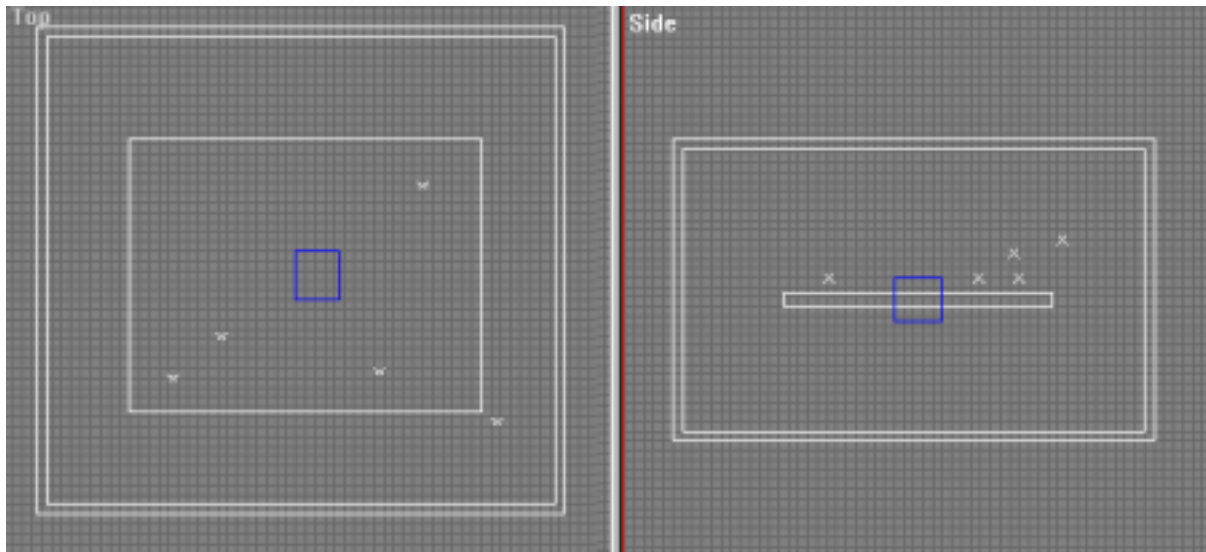
Du markierst **Solid** und **Cut Brush**.


Du kannst alle Abmessungen des Würfels einzeln vorgeben, einfacher ist es aber, du übernimmst die Standard-Werte und veränderst die Größe anschließend interaktiv in einem der Fenster. Du klickst also auf OK und ein Rechteck in blauer Farbe erscheint in allen Ansichten.


In der Werkzeugleiste kannst du mit dem Button  den Würfel in der Größe verändern und mit dem Button  kannst du ihn verschieben, bzw. rotieren, wenn die rechte Maustaste gedrückt wird.

Du schiebst und vergrößerst den blauen Würfel solange, bis er an der gewünschten Stelle den Fußboden schneidet.

Das müßte in etwa so aussehen:




Dann drückst du **Enter**. Jetzt ist der Würfel endgültig Teil der Levelgeometrie. Das 3D-Fenster aktualisierst du mit  und kannst jetzt tatsächlich durch den Fußboden schauen. Aber oh Schreck! Die Ränder sind nicht in der gleichen Textur, wie die Spielfläche. Du mußt also noch einmal Hand anlegen.


Dazu mußt du das Loch zuerst einmal selektieren. Du gehst in den Selectmode , und markierst in einer der Seitenansichten den neu eingefügten Cut Brush. Dann wählst du das Register **Textures**, und suchst dir eine passende Textur aus. Alternativ kannst du eine im 3D-Fenster sichtbare Textur anwählen, indem du Shift drückst und die linke Maustaste betätigst. Eine Textur ist jetzt gewählt, der Cut Brush ist selektiert, du drückst Apply (oder Return) und das Loch hat die richtige Textur.

Folgendes gibt es zu den Farben im Editierfenster zu sagen:

Dunkelblau sind Objekte, die erst dann Teil des Levels werden, wenn du Enter drückst, orange sind die Cut Brushes, weiß sind Entities und die additiven Brushes. Hellblau sind selektierte Objekte. Rosa sind selektierte Faces.

Als nächstes willst du einen Sammelstein in den Level einbauen?


Du drückst also wieder auf das  (neue Objekte generieren), und wählst das Register **Template**.

Neben dem Button  befindet sich eine Auswahlbox der SpinOff - Spielelemente. Dort wählst du actor\_rotatingStone, und drückst auf den „Lib“ Knopf.

Ein dunkelblaues Kreuz erscheint in den Editierfenstern. Du kannst es ebenso wie den Cut Brush von vorhin verschieben, jedoch nicht in der Größe verändern. Wenn du das Kreuzchen dort hast, wo es sein soll, drückst du wieder die Enter Taste.

Leider werden die aktiven Elemente des Spiels, die sogenannten **Entities** in den Fenstern des Editors nicht so dargestellt, wie es alle anderen Objekte tun. Deswegen siehst du z.B. statt eines Sammelsteines nur ein Kreuzchen.

Um zu erkennen, welches Kreuzchen welche Entity darstellt, gibt es den Entity Editor. Mit seiner Hilfe kannst du auch die Anzahl der für diesen Level zu sammelnden Steine um

eines erhöhen. Dazu drückst du auf  und aktivierst somit den Entity Editor. In der Auswahlbox wählst du „ChangeLevel1“ und setzt GoodiesToOpen ebenfalls auf „1“.

Mehr zum Entity Editor erfährst du im Kapitel Entity Eigenschaften.

Ich denke, du hast vorerst genug geändert, deswegen kompilierst du das Level, und schaust es dir wieder an. Vorher das Abspeichern nicht vergessen! Aber bitte unter einem anderen Namen!

Sollte der Compiler mit der Fehlermeldung „Leak !!! Level is not sealed“ stoppen, hast du irgend etwas außerhalb des Himmels plaziert, oder etwas „durchstößt“ den Himmel. Das ist nicht erlaubt, und muß korrigiert werden.

## 5 Die Spielelemente

Es gibt vier Klassen von Spielelementen: Akteure, Effekte, Magie und Objekte.

Einige dieser Spielelemente müssen in ihren konkreten Eigenschaften näher beschrieben werden. Dazu dienen sogenannte Entities (du erinnerst dich? Das sind alle aktiven Spielelemente wie Feuer, Magie, Kugeln usw.). Diese sind in den Editierfenstern als Kreuzchen markiert. Oft ist alleine die Position des Kreuzchens eine wichtige Eigenschaft der Entity: so sollte sich das Ziel natürlich an einer definierbaren Stelle im Level befinden – diese wird über die Position des Kreuzchens definiert. Allerdings muß man dem Ziel auch sagen, wie viele Sammelsteine es benötigt, um gelb zu werden.

Alle Spielelemente sind direkt über die „Lib“ Taste abrufbar. Einige sind so brandneu, dass es sie in bestehenden Levels noch nicht gibt. Diese Spielelemente sind mit (NEU !!) gekennzeichnet.

Leider entspricht nicht jedes Spielelement genau einer Entity, dazu sind manche Spielelemente einfach zu komplex. So besteht z.B. magic\_spikes aus zwei Entities (im Editor als Kreuzchen zu erkennen) und object\_simpleDoor aus einem Modell (die Tür) und einem Kreuzchen.

Trotzdem sind die Spielelemente *bevor* du sie mit Return zum Level hinzufügst als *ein* Kreuzchen in den Fenstern dargestellt, danach sind es mehrere Kreuzchen und/oder Objekte, die du alle markieren musst, wenn du das Spielelement noch einmal verschieben willst. Alternativ kannst du sie auch über ihre Gruppe selektieren.

Die folgende Liste zeigt dir alle verfügbaren Spielelemente. In geschweiften Klammern siehst du, welche Entities in welchem Spielelement Verwendung finden. Im Anhang sind alle Entities näher beschrieben. Dort kannst du nachschlagen, welche Eigenschaften verändert werden können, und was sie bewirken. Die meisten Eigenschaften sind allerdings bereits voreingestellt, so dass kaum Arbeit anfällt. Die Entities, bei denen du in jedem Fall noch mal Hand anlegen muß, sind **fett und kursiv** geschrieben. Das ist z.B. die Teleportation, bei der du das Ziel angeben musst, oder aber alle Sammelsteine, die per Hand durchnummeriert werden müssen, und ebenfalls einen Rematerialisierungsort benötigen.

Eine besondere Behandlung brauchen auch diejenigen Spielelemente, die AnimatedTextures bzw. ChaosTextures verwenden:

Willst du mehrere identisch animierte Texturen (z.B. 5 Lavalöcher) plazieren, so reicht nämlich eine ChaosTexture Entity aus. Kopiere nur den zugehörigen Brush, oder weise anderen Brushes die gleiche „Starttextur“ zu. Ein blinkender Pfeil und ein blinkender Streifen

sind allerdings zwei AnimatedTextures, da es sich um zwei unterschiedliche Textureffekte handelt.

## 5.1 Akteure

Akteure sind Kugeln und Sammelsteine.

actor_badMarble:	Eine böse Kugel, die nicht berührt werden darf.	{ <i>marble</i> }
actor_rotatingStone	Ein rotierender feststehender Sammelstein	{ <i>actor</i> }
actor_movingStone	Ein beweglicher Sammelstein (Berglevel)	{ <i>marble</i> }

## 5.2 Effekte

Alle graphischen Spielereien werden als Effekte bezeichnet. Auch hier gibt es eher harmlose Vertreter wie z.B. die blinkenden Pfeile im Übungslevel, aber auch Lavalöcher und Laserstrahlen. Willst du z.B. mehrere Lavalöcher plazieren, so reicht eine ChaosTexture Entity dafür aus. Kopiere nur den zugehörigen Brush, oder weise anderen Brushes die gleiche Starttextur zu.

effect_blinkingStripes	Blinkende Streifen wie im Zukunftslevel.	{AnimatedTexture}
effect_blinkingArrow	Blinkender Pfeil wie im Zukunftslevel.	{AnimatedTexture}
effect_lavaPit	Ein Lavaloch. Gefährlich für die Kugel.	{ChaosTexture}
effect_bubbles	Aufsteigende Luftblasen wie im Unterwasserlevel.	{effects}
effect_explosion	Explosionen. Gefährlich für die Kugel (NEU !!).	{effects, instantDeath}
effect_fireplace	Eine Feuerstelle.	{effects, dynamicLight}
effect_torch	Eine Fackel.	{effects, light}
effect_lightStandard	Ein Standardlicht zum Aufhellen der Region.	{light}
effect_lightSpot	Ein Spotlicht zum Aufhellen der Region.	{spotlight}
effect_laser	Ein Laserstrahl. Gefährlich für die Kugel.	{electricBolt, electricBoltTerminus}

## 5.3 Magie

Hier gibt's die magischen Kugeleigenschaften. Folgende Sorten von Magie werden derzeit von SpinOff unterstützt:

magic_big	Die Kugel wird groß.	{ effects, resize }
magic_small	Hier wird sie klein.	{ effects, resize }
magic_ill	Die Kugel wird krank (Ringlevel, Unterwasserlevel).	{ actor, design }
magic_heal	Hier wird die Kugel wieder geheilt.	{ effects, design }
magic_light	Die Kugel bekommt Licht.	{ effects, design }
magic_spikes	Die Kugel bekommt Spikes.	{ effects, design }
magic_transparent	Die Kugel wird durchsichtig.	{ effects, design }
magic_jump	Kugel kann mit linker Maustaste hüpfen ( <b>NEU !!</b> ).	{ effects, design }
magic_teleport	Kugel wird woanders hingebannt.	{ <i>beamField</i> }

## 5.4 Objekte

Ein paar Objekte wie Türen oder Plattformen, Eis usw. sind auch schon vorkonfiguriert.

object_bouncePad	Eine elastische Fläche, auf der die Kugel hüpfen kann.	
object_simpleDoor	Eine einfache automatische Tür.	{ door, ein Modell }
object_doubleDoor	Eine Doppeltür.	{ 2*door, 2 Modelle }
object_hospitalDoor	Eine Krankenhaustür.	{ door, ein Modell }
object_movingPlatform	Eine sich bewegende Plattform wie im Höllenlevel.	{ door, ein Modell }
object_movingObject	Ein Handle für allgemeine bewegte Objekte.	{ <i>door</i> }
object_ventilator	Ein Ventilator.	{ door, ForceField, ein Modell }
object_ice	Eine Eisfläche wie im Nordpollevel.	
object_sticky	Eine klebrige Fläche ( <b>NEU !!</b> ).	
object_friction	Eine Fläche mit erhöhter Reibung ( <b>NEU !!</b> ).	
object_floorLamp	Eine Bodenlampe.	{ light, corona }
object_fence	Ein Zaun.	
object_boxA	Eine Kiste.	
object_boxB	Noch eine Kiste.	


## 6 Fortgeschrittenes Leveldesign

Wie bereits erwähnt unterscheidet man zwischen dem Objekt (Brush) und seinen Oberflächen (Faces). Der Editor kennt somit auch zwei Betriebsmodi: Den zum Manipulieren von Brushes, und den zum Manipulieren von Faces.

### 6.1 Manipulation von Faces

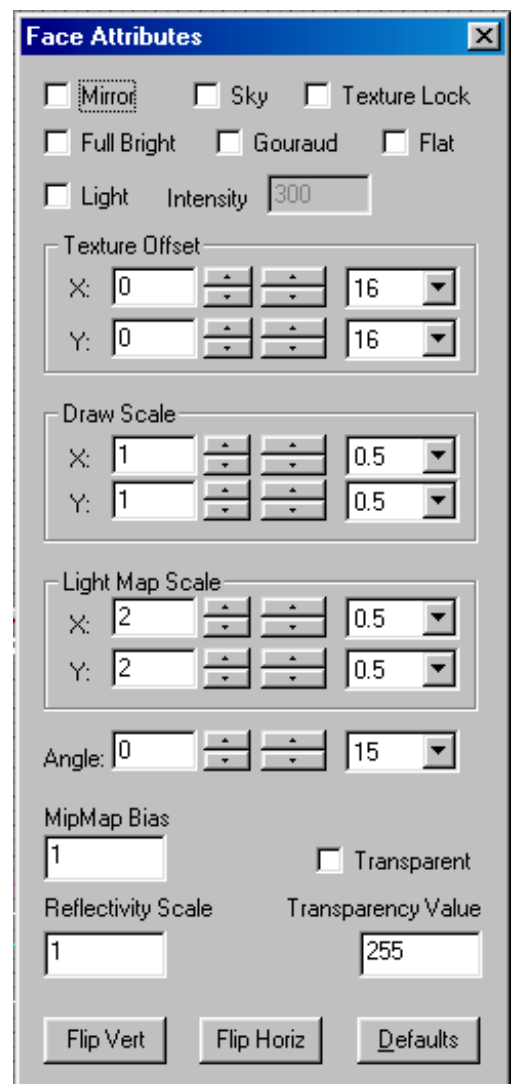
Vorhin hattest du das eingefügte Loch mit einer einheitlichen Textur versehen. Manchmal ist es sinnvoll, einzelnen Flächen (Sheets) unterschiedliche Texturen zu verpassen. Dabei gehst du wie folgt vor: Du markierst das Objekt (Brush) Und drückst die (Bild runter) Taste, oder im Menu Mode->Face Adjustment auswählen.

Der markierte Brush wird jetzt rosa und zeigt damit an, daß jetzt einzelne Flächen selektiert werden können. Dies machst du entweder durch selektieren mit der Maus im 3D-Fenster, oder du gehst sie der Reihenfolge nach durch, indem du die Cursortasten <- oder -> drückst. Nur die in rosa dargestellte Fläche kann jetzt mit der im Register „Textures“ gewählten Textur beklebt werden. Nachdem dies geschehen ist, kannst du die Textur noch an die Fläche anpassen. Wichtig ist das für Texturen mit Bildinformationen wie z.B. DragDoor2.

Wenn du jetzt  drückst erscheint der “FaceAttributes” Dialog. Mit ihm kannst du die Textur drehen, verschieben oder skalieren. Außerdem kannst du der Fläche spezielle Eigenschaften geben wie z.B. spiegelnd oder transparent.

TextureLock wählst du für bewegte Objekte wie z.B. Türen, damit die Textur auch dem Objekt folgt, wenn es sich bewegt.

Am besten kann man die Textur anpassen, wenn man sich im 3D Fenster direkt vor die Fläche stellt, so dass man sie gut sehen kann. Dann kann man mit den einzelnen Parametern spielen um zu sehen, was passiert.




Die Eigenschaften Mirror und Transparent kann man leider nur mit dem Levelpreviewer überprüfen. Transparent markieren, und z.B. ein Wert von 190 ergibt etwas halbtransparentes. Wenn du dann zusätzlich Mirror markierst, wird die Fläche spiegelnd. Aber Achtung: spiegelnde Flächen wirken sich sehr negativ auf die Bildwiederholrate aus.

Das Kästchen Sky kennst du ja fast schon: Hiermit wird nämlich die Levelumhüllende Skybox als solche gekennzeichnet. Dies ist im Startlevel für die “Skybox” bereits geschehen.

## 6.2 Manipulation von Brushes

Jetzt kannst du nachträglich einem Brush neue Eigenschaften zuweisen.

Dazu müssen du zuerst wieder in den “Brush Adjustment” Mode gehen. Im Menu also Mode->Brush Adjustment wählen.

Wenn du jetzt wieder  anklickst, öffnet sich der Brush Attributes Dialog. Dort kannst du z.B. einen soliden Körper zu einem Cut brush umfunktionieren, oder umgekehrt. Die anderen Typen sind: „Empty“ Brushes, die benötigst du z.B. für alles, was sichtbar, aber passierbar sein soll (auch für die Kamera). Im Gegensatz zu „Window“, das lichtdurchlässig, aber nicht passierbar ist. „Hint“ Brushes helfen dem Compiler, effizientere Level zu berechnen.

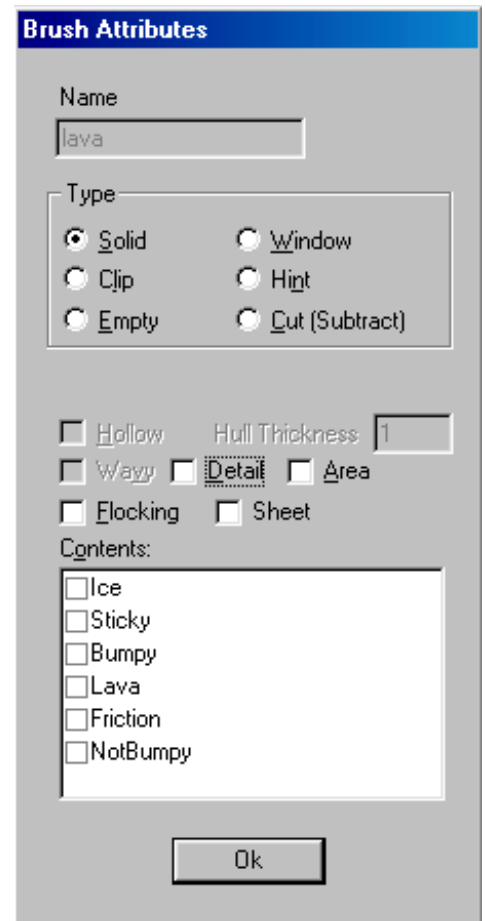
„Sheet“ sind zweiseitige Polygone, wobei die gelbe die sichtbare Seite ist (eignet sich für Bilder an der Wand, Spinnweben, Gitter etc.). Wenn du dir die Textur „gate“ anschaust (im Register Textures) siehst du ein schönes goldenes Tor vor rosa Hintergrund. Der rosa Hintergrund wird im eigentlichen Spiel durchsichtig, weil er mit der 256' ten Farbe der Farbpalette erstellt wurde.

In SpinOff gibt es mehrere Körper und Bodeneigenschaften, die du vorgeben kannst. Hiermit werden z.B. rutschige (Eis) oder klebrige Objekte erzeugt. Diese Eigenschaften kannst du über die Kästchen:

Ice, Sticky, Bumpy, Lava, Friction und/oder NotBumpy auswählen, und sie bedeuten genau das, was sie sagen.

Es können auch mehrere Eigenschaften für einen Körper gleichzeitig ausgewählt werden, wobei manche Kombinationen natürlich weniger sinnvoll sind (z.B. Ice+Lava). Entsprechend unvorhersehbar dürften die Resultate sein.

Die anderen Kästchen werden von SpinOff nicht ausgewertet.



## 6.3 Modelle

Bewegte Objekte werden im Editor Model genannt, und über das entsprechende Register verwaltet.


Modelle sind Türen, Aufzüge und sich bewegende Plattformen. Wir wollen jetzt eine sich bewegende Plattform erzeugen.

Dazu musst du zunächst einen statischen Brush in das Level einbauen und texturieren. (Bitte im Face Attributes Dialog das Texture Lock nicht vergessen!). Du selektierst den gewünschten Brush, und wechselst in das Model Register.

Dort drückst du Add Model. In den Mini-Dialog tippst du den gewünschten Namen ein.

Du kannst jetzt einen Bewegungsablauf für diese Plattform eingeben.

Den Rotationsmittelpunkt des Models kann man durch „Set Origin“ vorgeben; dabei dann nicht vergessen „done“ zu drücken. Rotierende Modelle funktionieren derzeit in SpinOff nur für Türen, nicht jedoch für Plattformen (wenn sich die Kugel hauptsächlich AUF dem Model befindet). Du brauchst also einen Drehpunkt für Plattformen und Aufzüge nicht zu definieren.


Nachdem du „Animate“ gedrückt hast, kannst du die Plattform mittels  an eine neue Position schieben oder drehen. Dann „stop Animation“ und den Zeitpunkt eingeben, zu dem das Model diese Position einnehmen soll (ab Zeitpunkt Null gerechnet). Diesen Vorgang solange wiederholen, bis der komplette Bewegungsablauf eingegeben ist. Es werden immer die Absolutzeiten ab Beginn der Animation gerechnet, keine Zeitdifferenzen!

Damit am Ende der Animation die Plattform nicht an den Anfang zurückspringt, definierst du auch den „zurück“ Bewegungsablauf. Ein sinnvoller Bewegungsablauf für eine Plattform sieht also folgendermaßen aus:

Position A	0	Ausgangsposition.
Position A	3	Keine Bewegung: Die Plattform ruht für 3 Sekunden (zum Betreten).
Position B	6	Nach 3 Sekunden ist die Zielposition erreicht.
Position B	9	Keine Bewegung. Die Plattform ruht für 3 Sekunden auf Zielposition.
Position A	12	Die „zurück“ Bewegung dauert 3 Sekunden.

Soll das Modell eine einmalige Bewegung ausführen, und dann dort stehenbleiben, dann definieren keine „zurück“ Bewegung, sondern fügen zum Schluß eine „Nullbewegung“ mit sehr großem Zeitpunkt ein. Also z.B.:

Position A	0	Ausgangsposition.
Position A	3	Dort bleibt die Plattform für 3 Sekunden (zum Betreten).
Position B	6	Nach 3 Sekunden ist die Zielposition erreicht.
Position B	10000	Die Plattform verbleibt auf der Zielposition.

Jetzt musst du noch ein paar Eigenschaften dieser Plattform definieren. Dazu gibt es das Spielelement „object\_movingObject“, das eine Entity des Typs „Door“ erzeugt. Dieses platzierst du möglichst in der Plattform, damit du sie später einfacher wiederfindest, und gibst im Entity Editor  folgende Daten ein:

- Model                      Hier wählst du das eben erstellte Modell aus.
- AdditionalModel          Hier gibst du nichts ein.
- AutoDoor                    0 (das Modell wird nicht von der Kugel angestoßen, sondern
- LoopedMotion              1 (das Modell führt zyklisch immer wieder die gleiche Bewegung aus).
- BadMarbleTrigger        0
- GoodMarbleTrigger      0

SpinOff rundet Bewegungsabläufe ab. Wem dies nicht gefällt (Es erschwert das Betreten und Verlassen einer Plattform), der muss vor und nach jedem Bewegungsschritt eine zusätzliche „Nullbewegung“ einführen, mit ganz kurzer Zeitdauer. Also z.B. folgendermaßen:


Position A	0	Ausgangsposition.
Position A	3	Keine Bewegung. Dort bleibt die Plattform für 3 Sekunden (zum Betreten).
Position A	3.1	Keine Bewegung, um das Abrunden von Bewegungsabläufen zu verhindern.
Position B	6	Nach 2.9 Sekunden ist die Zielposition erreicht.
Position B	6.1	Keine Bewegung, um das Abrunden von Bewegungsabläufen zu verhindern.
Position B	9	Keine Bewegung, die Plattform verbleibt 3 Sekunden (zum Verlassen).
Position B	9.1	Keine Bewegung um das Abrunden von Bewegungsabläufen zu verhindern.
Position A	12	Die „zurück“ Bewegung dauert 2.9 Sekunden.

## 6.4 Beleuchtung

Die Atmosphäre eines Levels entsteht maßgeblich über die Beleuchtung. Bisher kam alles Licht über die „default light level“ des Compilers. Für das StartLevel stehen dort drei Zahlen, für die allgemeine Helligkeit Rot, Grün und Blau. Setzen du diese drei Zahlen zu null, oder einfacher: du deaktivierst „default light level“ wird es (zumindest im Levelpreviewer) stockfinster.

Alle Arten von Beleuchtung gehören zu den „Entities“ und werden somit im Template-Register ausgewählt. Schau dir der Einfachheit halber erst einmal das „normale“

Licht an. Dazu pickst du dir aus dem Spielelemente Menü den Eintrag „effect\_lightStandard“ heraus, und setzt das dann auftauchende Kreuz in das Level (Return nicht vergessen).

Dann öffnest du den „Entity“-Dialog  und wählst die eben plazierte Licht Entity „light1“ aus, stellst „light“ auf 400, und suchst dir auf Wunsch eine passende Farbe aus. Im Levelpreviewer kannst du es dir anschauen, nachdem du dein Level kompiliert hast, aber bitte nicht vergessen „Default Light Level“ im Compiler Dialog zu deaktivieren. Dies ist die einfachste Form von Licht, die du produzieren kannst.

„Effect\_spotLight“ hat eine gerichtete Form und erzeugt eine Entity mit Namen SpotLight. Man kann in den Editierfenstern die Strahlungsrichtung (Move/Rotate) spezifizieren, und im „Entity“ Dialog den Öffnungskegel.

„SunLight“ ist die einfachste Art ein Level komplett zu beleuchten (abgesehen vom default Light im Compiler Dialog), es nimmt in der Stärke nicht ab, und ist immer parallel. Die Positionierung der Entity spielt somit keine Rolle.

Allerdings strahlt SunLight nicht sehr tief in die Gänge und Räume z.B. einer Pyramide hinein. Es simuliert eben echtes Sonnenlicht.

Dynamische Lichter gibt es mit dem Spielelement „effect\_fireplace“. Hier wird eine Entity „DynamicLight“ erzeugt, mit dem sich schon wesentlich bessere Effekte erzielen lassen. Aber Achtung, dynamische Lichter wirken sich sehr unangenehm auf die Bildwiederholrate aus. Außerdem ist ihre Gesamtzahl auf 29 begrenzt.

Du plazierst einen Fireplace im Level, und nimmst im Entity-Dialog die Feineinstellungen vor, also Eintrag „DynamicLight1“ auswählen, und z.B. in folgende Einstellungen ändern:

Color:	Ein helles Orange
RadiusFunction:	gztgem
RadiusSpeed:	6.7

Diese ergeben ein unregelmäßig flackerndes oranges Licht, das z.B. einem offenen Feuer entstammen könnte. Die Funktion 'RadiusFunction' bestimmt dabei die Licht-Intensität des Flackerns, 'a' ist sehr dunkel und 'z' ganz hell. Der String gibt an, in welcher Reihenfolge welche Lichtintensitäten abgestrahlt werden.

Um das ganze etwas ruhiger zu gestalten, kann man ein normales Licht dazugeben, das für eine gewisse Grundhelligkeit sorgt.

Man kann die „Dynamische Lichter“ auch an Modelle binden, um z.B. Fackeln an einer bewegten Plattform oder ähnliches zu bewerkstelligen. Dazu öffnet man den Entity-Dialog des dynamischen Lichts, und weist im Feld „Model“ das entsprechende Modell zu. Dann bewegt sich das Licht im Einklang mit dem Modell. Die Position des dynamischen Lichts sollte irgendwo auf der Plattform Grundposition (time=0) plaziert sein.

## 6.5 Neue Texturen

Will man ein Level mit neuen Texturen erstellen, so gibt es zwei Möglichkeiten: Selber welche malen, oder sich (freie!) Texturen besorgen. 4fo hat einen Downloadbereich, wo es freie, SpinOff-taugliche Texturen gibt.

Nicht jede beliebige Textur kann in SpinOff verwendet werden.

Zuersteinmal muß die Textur eine Größe von 16\*16, 32\*32, 64\*64, 128\*128 oder maximal 256\*256 Pixel haben. Dann dürft ihr nur eine Farbpalette von maximal 256 Farben verwenden. Die Bitmap muß das \*.bmp Format haben. Wollt ihr transparente Elemente in der Textur haben (wie z.B. die Textur „Gate“), so muß die transparente Farbe auf dem Paletteneintrag 256 liegen. So etwas kann man z.B. mit dem Shareware Malprogramm PaintShop machen.

Wie bekommst du nun die Texturen in den Editor? Dazu gibt es das Hilfsprogramm Tpack.

Startet also einmal Tpack. Dann die Datei Marble.txtl öffnen (liegt im art Verzeichnis). Da sieht man alle derzeit verfügbaren Texturen. Neue Texturen werden hinzugefügt, indem man die (taugliche!) Textur im Explorer markiert und in den Arbeitsbereich von Tpack hinüberzieht. Dann wieder abspeichern, SpinnOffEdit erneut öffnen und schon sind die neuen Texturen verfügbar. Ein bekannter Bug von Tpack ist allerdings, dass es Bitmaps beim extrahieren um 180° dreht. Wenn ihr also Bitmaps extrahiert, und dann wieder importiert, sind in bestehenden Levels die entsprechenden Texturen auf den Kopf gestellt.

## 6.6 Ein neuer Himmel

Sehr wichtig für das Ambiente eines Levels ist auch der Himmel. Im Startlevel ist bereits ein Sky voreingestellt, den du aber über das Register Sky ändern kannst. So ist es möglich, z.B. eine Sternentextur zu verwenden. Diese muss dann für alle sechs Seiten des „Himmelswürfels“ eingestellt werden.

SpinOff wertet die Rotations- und Texturescale Felder nicht aus.

Die Erstellung eines gänzlich eigenen Himmels ist nicht ganz einfach, da der Himmel einen sphärischen (runden) Eindruck hinterlassen soll, der HimmelsBrush jedoch quadratisch ist. Wählt man eine normale Textur, so wirkt das Himmelsgewölbe ebenfalls quadratisch, was unschön aussieht.

Man braucht also Bilder, in denen eine entsprechende Transformation bereits durchgeführt wurde.

Ich habe neben Sternen und Wolken auch Landschaften als Skytextur eingesetzt, die ich mit TerraGen erstellt habe. TerraGen ist Shareware, und bis Version 0.6 auch für kommerzielle Zwecke kostenlos verwendbar (Du willst doch dein Level verkaufen, oder?). TerraGen erzeugt sphärisch wirkende Würfelt Texturen wenn man die Kamera jeweils um 90° dreht, und einen „Zoom“ von 1 wählt.

Dann gibt es noch etliche kommerzielle Programme, zu denen ich aber nicht viel sagen kann.

Die neu erstellten Skytexturen (6 Stück) müssen ebenfalls in eines der üblichen Texturformate gewandelt werden (siehe auch das Kapitel „neue Texturen“), die am besten mit 256\*256 Pixel und über Tpack importiert werden.

## 6.7 Compiler Optionen

Eine wichtige Eigenschaft des Compilers kennt ihr ja schon: Default Light. So kann man am schnellsten ein funktionsfähiges Level bauen. Mit der Einführung von Lichtquellen kann man Default Light deaktivieren. Dann gibt es aber noch weitere „light options“:

Mit Radiocity wird das Level erst richtig gut beleuchtet. Leider dauert dann das Kompilieren bei großen Levels besonders lange. Ich würde deswegen empfehlen, Radiocity erst zum Schluß dazuschalten, und nur eine Feinabstimmung der Lichtverhältnisse durchzuführen. ExtraSamples und FastPatch bringen nicht viel zusätzlichen optischen Gewinn bzw. Kompilierzeit. Wenn ihr Preview markiert, startet sich im Anschluß an den Kompilervorgang der Levelpreviewer automatisch. Mit den anderen Settings habe ich nicht gearbeitet.

## 6.8 Der Levelpreviewer

Im Levelpreviewer gibt es zwei zusätzliche Features, die es in SpinOff nicht gibt:

Screenshots, sowie eine FPS Anzeige.

Mit jedem Tastendruck „s“ wird ein Screenshot ins Verzeichnis art/Screenshots abgelegt.

Die Namensvergabe erfolgt automatisch: „Bmpx.bmp“ in der gewählten SpinOff Auflösung, x zählt von 1 an immer weiter hoch.

Achtung, wenn du SpinOff neu startest, werden die alten Bilder beginnend bei 1 wieder neu überschrieben. Deswegen musst du die interessanten Bilder vor einem Neustart retten indem du sie umbenennst.

Mit Tastendruck „f“ werden in der linken oberen Ecke einige statistische Daten zum Level angezeigt, unter anderem die Bildwiederholrate. Diese sollte so ausgelegt sein, dass dein Level auch auf älteren Rechnern funktioniert. Ich habe dazu den D3D Treiber (d3ddrv.dll) umbenannt, so dass SpinOff mit dem (langsamen) Softwaretreiber arbeiten muss. Ich habe versucht die Levels dann so klein zu halten, (bzw. mit wenig Schnickschnack) dass sie noch mit durchschnittlich über 10 Bildern pro Sekunde bei großer Bildschirmauflösung (1024 \* 768) funktionieren.

Mit Restart Game kann man jederzeit wieder von vorne anfangen, was aber nicht unbedingt notwendig ist, da der Lebenszähler nicht herunter zählt.

Der Levelpreviewer startet nur in einem Fenster, egal was ihr in den SpinOff Settings einstellt. Das ist für Windows die stabilere Einstellung.

## 7 Tipps

Bastelt nicht einfach drauf los! Überlegt Euch zuerst ein Thema für das Level, dann eine Bleistiftskizze über Orte, Magie und Gefahren, dann Realisiert das ganze. Dieses Vorgehen ist das Effektivste.

Bitte verändert den Charakter des Spiels nicht! So wäre es doch sehr unschön, wenn die Spike-Magie an sich drehenden Seifenblasen (eigentlich Transparenz) gekoppelt wäre.

Wenn der Boden aus mehreren Brushes besteht, so müssen die Flächen absolut plan aufeinander stoßen. Schon kleine Höhenunterschiede lassen die Kugel an der Kante abprallen. Das gilt auch für Banden.

Zur Erleichterung kann man im Editor „Snap to grid“ wählen. Funktioniert aber leider nicht immer exakt, vor allem nicht bei gedrehten Brushes.

Wollt ihr eine Steigung einbauen, so muß auch diese ohne Stufe erreichbar sein.

Arbeitet mit möglichst wenigen Cut Brushes, sie bereiten in großer Zahl angewendet große Probleme.

Vermeidet sehr spitzwinklige Anordnung von Brushes. Ca 45° dürfe die Obergrenze sein, sonst kann die Kugel im spitzen Winkel durch die Wand rollen.

Zu viele nacheinander durchgeführte Modifikationen an den Brushes (copy operationen, mehrfaches vergrößern und verkleinern, mehrfaches drehen) führt zu schwarzen Flecken im Level, die nicht mehr zu beseitigen sind. Deswegen lieber häufiger Zwischenstände abspeichern und immer kontrollieren, ob es noch gut aussieht.

Der Editor kennt kein Undo!!! Auch das sollte ein Grund sein, häufig Zwischenstände abzuspeichern.


## 8 Anhang Entity Eigenschaften

Jetzt sieh dir die Entities, aus denen die Spielelemente bestehen einmal näher an, weil du nur so die konkreten Eigenschaften deines Levels bestimmen und verändern kannst. Eigenschaften, die du in jedem Fall definieren muß (wo eine pauschale Voreinstellung keinen Sinn macht), *sind kursiv und fett* geschrieben.

Hier habe ich alle derzeit überhaupt möglichen Entities aufgelistet. Wenn eine hier beschriebene Entity im Editor nicht erscheint, ist das kein Grund zu Besorgnis: du hast einfach noch kein Spielelement plaziert, das die entsprechende Entity enthält.

Leider lassen sich mit diesen Entities nicht alle der bekannten Levels erzeugen (z.B. die Fische im Unterwasserlevel). Das hat zwei Gründe: Erstens gibt es noch organisatorische Probleme, mehr als ein bsp-File als Level upzuloaden (deswegen kannst du auch noch kein

eigenes MIDI File als Musik beilegen), zweitens wird es dann gleich sehr viel komplexer. Es wird aber für alle erfolgreichen Leveldesigner in absehbarer Zeit ein kleines Upgrade geben, mit dem auch dies möglich sein wird.

Drücke  in der Werkzeugleiste. Der Entity Editor (Properties Dialog) erscheint, und zeigt dir alle bisher im Level vorhandenen Entities an.

In der oberen Auswahlbox kannst du dir die zu verändernde Entity aussuchen.

## 8.1 PlayerStart

Dies ist eine Entity, die nur einmal im Spiel vorkommen darf. Sie ist bereits Teil des StartLevels, steuert aber einige interessante Eigenschaften des Spiels. Hier kannst du folgende Parameter verändern:

- Distance      Das ist die Entfernung zwischen dir und deiner Spielkugel während des Spiels.
- Orientation      Der Blickwinkel, unter dem du auf die Spielkugel schaust. Besser ist es, diese interaktiv in den Editierfenstern einzustellen. Da ist ein wenig herumprobieren angesagt, um eine bessere, als die Default Einstellung zu finden.
- Gravity      Die Schwerkraft. Je größer sie wird, desto schwieriger wird die Steuerung der Kugel. Negative Werte ziehen die Kugel nach unten, positive nach oben (wie im Unterwasserlevel).

## 8.2 ChangeLevel

Auch dies ist eine Entity, die nur einmal im Spiel vorkommen darf. Sie ist bereits Teil des StartLevels und stellt das zu erreichende Ziel des Levels dar. Wo du sie hinsetzt, sieht man nachher das rote, bzw. gelbe Tor. Folgende Parameter sind manipulierbar:

- GoodiesToOpen**      Die Anzahl der Sammelsteine, die du brauchst, um das Tor zu öffnen.
- Model      Du kannst das ChangeLevel auf ein bewegliches Modell setzen. Dann bewegt es sich mit diesem. **(NEU !!)**
- Origin      Die Koordinaten von Levelchange. Einfacher läßt sich Levelchange im den Editierfenstern platzieren.

## 8.3 Artefact

Das sind die feststehenden, sich drehenden Sammelsteine. Bitte unbedingt auf die Nummerierung in „Number“ achten.

- Origin Die Position des Sammelsteins.
- Number** Alle Sammelsteine eines Levels müssen durchnummeriert werden !!! Jeder Sammelstein muß eine eindeutige Nummer von 1-20 tragen.  
Sammelsteine, welche die Number=0 haben, sind nach dem Verlust einer Kugel wieder vorhanden.
- Model Du kannst den Sammelstein auf ein bewegliches Modell setzen. Dann bewegt er sich mit diesem (NEU !!)

## 8.4 Marble

Das sind alle rollenden Gegenstände im Spiel. Das kannst sein: Die Spielerkugel, rollende Sammelsteine und böse Kugeln. Die Spielerkugel ist bereits Teil des StartLevels. Mit dieser Entity kann keine weitere Spielerkugel erzeugt werden!!! Unter anderem deswegen ist StartLevel schreibgeschützt!

- Origin Die Startposition der Kugel.
- NewOrigin** Sammelsteine und böse Kugeln können nach ihrem Tod wieder erstehen. Hier musst du die Koordinaten für diesen Ort eingeben. Soll die Kugel wieder dort entstehen, wo sie sich beim Levelstart befand, übernimmst du die Werte aus Origin. Steht hier Blödsinn, so kann ein Level unter Umständen nicht beendet werden, oder SpinOff beendet mit Fehler.
- ActorName Sammelsteine heißen „Diamond.act“, böse Kugeln „marble\_bad.act“ harmlose Kugeln heißen „marble.act“. Andere Kugeln werden derzeit nicht unterstützt.
- Number** Falls die Kugel ein Sammelstein ist, muß auch dieser wie oben beschrieben durchnummeriert werden. Die Nummern müssen für alle Sammelsteine (drehende und rollende) eindeutig sein.
- Goodie Für einen Sammelstein muß hier eine 1 stehen, sonst 0.
- Bad Für eine böse Kugel muß hier eine 1 stehen, sonst 0.

## 8.5 Music

Das ist die Musik zum Level. Derzeit können nur im art Verzeichnis bestehende midi Files verwendet werden.

- MidiFile z.B. „horror.mid“.

## 8.6 Design

Das ist die Magie-Entity, die der Kugel ein neues Design, und somit eine neue Eigenschaft gibt.

Mehrere magischen Eigenschaften von Design lassen sich nicht in einer Kugel kombinieren. Ausnahme: Die Kugel kann Licht ausstrahlen, und gleichzeitig eines der anderen Designs annehmen. Design nur zusammen mit dem entsprechenden Effekt plazieren!

- Origin Die Position der Design Magie.
- Model Du kannst die Magie auf ein bewegliches Modell setzen. Dann bewegt sie sich mit diesem (**NEU !!**).
- NewDesign Die neue Eigenschaft. Folgende gibt es:  
0=Normal, 1=durchsichtig, 2=Spikes, 3=Krank, 4=Licht, 5=Jump
- Time Die Zeitdauer, solange die Magie anhält.

## 8.7 Resize

Auch eine Magie: Größenänderung der Kugel. Die Größenänderung läßt sich mit der Design Magie kombinieren. Resize nur zusammen mit dem entsprechenden Effekt plazieren.

- Origin Die Position der Resize Magie.
- Model Du kannst die Magie auf ein bewegliches Modell setzen. Dann bewegt sie sich mit diesem (**NEU !!**).
- NewSize Die neue Größe. Groß=20, klein=6, normal=12.
- Time Die Zeitdauer, solange die Magie anhält.

## 8.8 Door

Diese Entity steuert alle beweglichen Objekte. Also Türen, Plattformen, Aufzüge.

- Model Du musst zuerst ein Modell mit Bewegungsablauf definiert haben (Kapitel Models). Wenn dies geschehen ist, kannst du es hier auswählen.
- AdditionalModel Ein weiteres Modell, das sich bewegt, wenn obiges Modell berührt wird. Wird z.B. für Flügeltüren benötigt.
- AutoDoor Bewegt sich bei Berührung=1, Krankenhaustür=2.
- LoopedMotion Wenn 1, dann bewegt sich das Model zyklisch immer wieder (Höllenlevel).
- BadMarbleTrigger Wird von bösen Kugeln angestoßen. (**NEU !!**)
- GoodMarbleTrigger Wird von guten Kugeln angestoßen.

## 8.9 InstantDeath

Das ist die Entity für Explosionen. Die Explosion erfolgt alle 2 sec. InstantDeath nur zusammen mit dem entsprechenden Effekt plazieren.

- Origin Die Position der „Todeszone“.
- Model Du kannst die Todeszone auf ein bewegliches Modell setzen. Dann bewegt sie sich mit diesem (NEU !!).

## 8.10 Effects

Hiermit werden alle grafischen Effekte wie aufsteigende Luftblasen, Fackeln, Explosionen usw. realisiert. Viele Effekte werden zusammen mit anderen Entities plaziert (z.B. Design, InstantDeath). Die korrekte Zuordnung ist in den Bibliothekselementen bereits gegeben. Bitte in diesen Fällen den Effekt nicht von der zugeordneten Entity trennen.

- Origin Die Position des Effekts.
- Model Du kannst die Magie auf ein bewegliches Modell setzen. Dann bewegt sie sich mit diesem (NEU !!).

## 8.11 AnimatedTexture

Definiert wechselnde Texturen (z.B. effect\_blinkingStripes, effect\_blinkingArrow).

Dabei wird die Textur des plazierten Brushes periodisch durch andere Texturen ersetzt. Wenn eine AnimatedTexture Entity verwendet wird, ersetzt sie ALLE gleichnamigen Texturen des Levels. Das bedeutet, daß du zur Erzeugung von z.B. mehreren blinkenden Pfeilen nur eine AnimatedTexture Entity brauchst, und alle gewünschten Brushes mit der Textur „steel4arrow“ versiehst.

Das zur Entity gehörende Kreuzchen läßt sich nicht verschieben. Das ist aber auch egal.

## 8.12 ChaosTexture

Definiert veränderbare Texturen (effect\_lavaPit).

Dabei wird die Textur „lava“ einem Veränderungsprozess unterzogen. Wenn eine ChaosTexture Entity verwendet wird, ersetzt sie ALLE gleichnamigen Texturen des Levels. Das bedeutet, daß du zur Erzeugung von mehreren Lavalöchern nur eine ChaosTexture Entity brauchst, und alle gewünschten Lava Bereiche mit der Textur „Lava“ versiehst.

Das zur Entity gehörende Kreuzchen läßt sich nicht verschieben. Das ist aber auch egal.

## 8.13 Corona

Simuliert den Strahlenkranz einer Lichtquelle.

- Origin Die Position der Corona.
- Model Du kannst die Corona auf ein bewegliches Modell setzen. Dann bewegt sie sich mit diesem. **(NEU !!)**
- MaxRadius Die Größe der Corona.
- Color Die Farbe der Corona.

## 8.14 BeamField

Wenn die Kugel in ein BeamField rollt, wird sie an eine andere Stelle teleportiert.

- Origin Die Position des BeamFields.
- Model Du kannst das BeamField auf ein bewegliches Modell setzen. Dann bewegt es sich mit diesem. **(NEU !!)**
- NewPos* Hier die neue Position der Kugel.

## 8.15 ElectricBolt

Eine Laserbarriere. Wenn Die Kugel nicht unsichtbar ist (oder drunter durchpasst), verbrennt sie darin.

Eine Laserbarriere liegt zwischen zwei Punkten, der eine wird mit dieser Entity definiert, der andere mit ElectricBoltTerminus.

- Origin Die Startposition der Laserbarriere.
- Terminus Die Endposition der Laserbarriere

## 8.16 ElectricBoltTerminus

Das andere Ende der Laserbarriere

- Origin Die Startposition der Laserbarriere.
- Model Die Endposition kann sich mit einem Modell bewegen. **(Neu !!)**

## 8.17 ForceField

Ein Kraftfeld, das die Kugel in eine Richtung drückt (z.B. ein Luftstrom).

- Origin Die Position des ForceField.
- Model Das ForceField kann sich mit einem Modell bewegen (**NEU !!**).
- Direction Richtung, in die das Kraftfeld wirkt, Stärke in X, Y, Z-Richtung.
- Range Bereich um den Ursprung des Kraftfeldes, in dem es wirkt.

## 8.18 Light

Statisches Licht (beleuchtet nur die statischen Teile des Levels)

- Origin Die Position des Lichts.
- Color Farbe des Lichts.
- Light Lichtstärke.
- Style 0=Licht. 1=“Dunkelquelle.“

## 8.19 Spotlight

Statisches Licht, das die Beleuchtungscharakteristik eines Spot-Scheinwerfers hat

- Origin Die Position des Lichts.
- Color Farbe des Lichts.
- Light Lichtstärke.
- Style 0=Licht. 1=“Dunkelquelle“.
- Arc Öffnungswinkel.
- Angles Richtung des Spots.

## 8.20 Sunlight

Dies ist eine Entity, die nur einmal im Spiel vorkommen darf. Sie ist bereits Teil des StartLevels. Sie stellt gerichtetes, paralleles Sonnenlicht dar. Willst du kein Sunlight, so setze die Lichtstärke zu 0.

- Color Farbe des Sonnenlichts.
- Light Lichtstärke.
- Angles Richtung des Sonnenlichts.
- Style 0=Licht. 1=“Dunkelquelle.“

## 8.21 DynamicLight

Hier kann die Eigenschaften von dynamischem Licht definiert werden. Dynamisches Licht kann seine Helligkeit ändern, und beleuchtet auch die Kugel. Es darf nicht mehr als 29 dynamische Lichter in einem Level geben.

- Origin Die Position der Lichtquelle.
- Model Du kannst die Lichtquelle auf ein bewegliches Modell setzen. Dann bewegt sie sich mit diesem. **(NEU !!)**
- RadiusFunction Hier wird das „Geflacker“ der Lichtquelle definiert. a=dunkel, z=hell.
- RadiusSpeed Wie schnell das Licht die Helligkeitswerte von RadiusFunction durchlaufen soll.
- Color Farbe der Lichtquelle.

## 8.22 Camera

Das ist eigentlich keine Entity von SpinOff, sondern eine Funktion des Editors. Sie zeigt an, wo sich gerade die „Kamera“ für das 3D Fenster befindet.

## 9 Anhang Begriffe

Einige der Fachbegriffe die in diesem Dokument verwendet werden kannst du hier nachschlagen.

akteure	Akteure sind Kugeln und Sammelsteine.
angle	Winkel. Im 3D Editor zumeist mit drei Koordinaten spezifiziert.
brush	Ein geometrisches Objekt mit Volumen.
cut brush	Subtraktive Objekte. Mit ihnen kann man z.B. Löcher in den Boden schneiden.
effekte	grafische effekte wie z.B. blinkenden Pfeile, Laserstrahlen und Luftblasen.
entities	Aktive. Mit ihnen realisiert man Kugeln, Sammelsteine, grafischen Effekte.
face	Die Seitenflächen eines brushes. Sie tragen die Textur.
hollow brush	Ein brush mit innerem Hohlraum. Wird zur Erzeugung von Räumen benötigt.
magie	Die magischen Kugeleigenschaften.
model	bewegte geometrische Objekte. Zum realisieren von Türen oder Plattformen.
objekte	Alle aus brushes bestehender Dinge, z.B. eine Box, oder eine Tür.
origin	Ursprung, Position. Im 3D-Editor mit drei Koordinaten spezifiziert.
sheet brush	Ein brush, der nur eine sichtbare face besitzt.
sky	Der Himmel. Ein geschlossener (würfelförmiger) Raum, der ALLES umgibt.
solid brush	Ein brush ohne inneren Hohlraum.
Spielelement	Es gibt vier Klassen von Spielelementen: Akteure, Effekte, Magie und Objekte.
template	Vorlagen zum Leveldesign (fertige Spielelemente und Objekte).
textur	Eine Bitmap zum „bedrucken“ von faces.