

HANS-GEORG SCHUMANN

SPIELE PROGRAMMIEREN MIT

UNITY

FÜR KIDS

2. AUFLAGE

MIT SPASS 3D-SPIELE
SELBST ENTWICKELN



mitp

INHALT

EINLEITUNG	11
Welche Werkzeuge benötigen wir?	12
Was bietet dieses Buch?	13
Wie arbeite ich mit diesem Buch?	13
Was brauchst du für dieses Buch?	14
Wie gut kennst du C#?	15
Hinweise für Lehrer	16
DAS ERSTE PROJEKT	17
Unity starten	18
Ein Objekt zum Spielen	22
Gravitation und Kollision	31
2D oder 3D?	37
Unity beenden	43
Zusammenfassung	45
Ein paar Fragen	46
... und eine Aufgabe	46
SCRIPT-PROGRAMMIERUNG	47
Ein Script erstellen	47
Klassen und Methoden	54
if-Strukturen	57
Schubsen oder schieben?	61
Mal schwerelos, mal »bouncy«	64
Import und Export	68
Zusammenfassung	73
Ein paar Fragen	74
... und eine Aufgabe	75

1

2

3	EINE FIGUR ZUM SPIELEN	77
	Ein neues Spielobjekt	78
	Bilder fürs Sprite	81
	Ein Script für die Figur	84
	Character Controller	88
	Material und Textur	93
	Zusammenfassung	101
	Ein paar Fragen	102
	... und eine Aufgabe	102
 4	JUMP & RUN	 103
	Steuersystem	104
	Das richtige Bild	110
	Eine eigene Methode	115
	Laufen, Springen, Schubsen	118
	Bouncy Ball	121
	Trigger	123
	Texturen	127
	Zusammenfassung	130
	Ein paar Fragen	131
	... und ein paar Aufgaben	132
 5	SIGHTSEEING IN 3D	 133
	Einfache Szene in 3D	134
	Bewegte Kamera	138
	Springen und Drehen	140
	Player mit Kamera	144
	3rd oder 1st Person?	148
	Fertig-Player aus der Packung?	152
	Zusammenfassung	156
	Ein paar Fragen	157
	... und ein paar Aufgaben	157
 6	LANDSCHAFTEN	 159
	Von der Ebene zum Terrain	160
	Ein Gelände gestalten	163
	Rundgang und Asset-Suche	167
	Landschaftspflege	174
	Vegetation	178

Noch mehr Details?	183
Zusammenfassung	188
Ein paar Fragen	189
... aber keine Aufgabe	189
ERDE, WASSER, LUFT	191
Auf und ab	192
Grenzkontrollen	195
Wind	199
... und Wasser	202
Entschlackungskur	208
Kugel mit RigidBody	209
Kollision mit Folgen	212
Zusammenfassung	215
Ein paar Fragen	216
... und eine Aufgabe	216
BAUWERKE	217
Baumaterial	217
Platten legen	222
Prefab-Transport I	229
Prefab-Transport II	235
Innenansichten	237
Steigungen	241
Zusammenfassung	244
Ein paar Fragen	245
... und ein paar Aufgaben	245
KLETTERN UND SCHWIMMEN	247
Ein Kletter-Trigger	247
Der Player lernt klettern	250
Ein kleiner Schubs	253
See-Landschaft	257
Unterwasser-Atmosphäre	261
Waten, Schwimmen, Tauchen	264
Bewegungskontrolle	269
Zusammenfassung	271
Keine Fragen	272
... aber ein paar Aufgaben	272

7**8****9****7**

10	ANIMATION UND NAVIGATION	273
	Ein kleines Monster	274
	Animator und Keyframes	277
	Das »Ding« bewegt sich	282
	Kleiner Probelauf	287
	Ein Navigator für die Kreatur	290
	Verfolgung an, Verfolgung aus	297
	Hindernislauf	300
	Zusammenfassung	303
	Ein paar Fragen	304
	... und eine Aufgabe	304
11	SPIEL MIR DAS LIED VOM TOD	305
	Angriff und Verteidigung	306
	Tödliche Kugeln	310
	Animationen organisieren	312
	Stehen – Gehen – Sterben	319
	Tod des Players?	325
	Die Kreatur wird zum Monster	326
	Zusammenfassung	333
	Ein paar Fragen	334
	... aber nur eine Aufgabe	334
12	STRAHLEN, PARTIKEL UND SOUND	335
	Raycasting	335
	Todesstrahlen	339
	Partikelsysteme	343
	Flammenwerfer	350
	Schrittgeräusche	352
	Noch mehr Sound?	356
	Zusammenfassung	359
	Ein paar Fragen	360
	... und ein paar Aufgaben	360
13	GAME TUNING	361
	Die Kreatur rüstet auf	362
	Gesundheits-Balken	364
	Energiekontrolle für den Player	371
	... und für die Kreatur	374

INHALT

Game Over	376	
Aufmarsch der Gegner	378	
Play the Game	382	
Zusammenfassung	388	
Keine Frage und keine Aufgabe mehr	389	
ANHANG A: UNITY INSTALLIEREN	391	A
ANHANG B: DEBUGGING	405	B
STICHWORTVERZEICHNIS	409	



Eigene Fantasiewelten erschaffen, in denen man sich frei bewegen kann. Selbst gebauten Figuren begegnen. Abenteuer selbst erfinden, den Verlauf von Ereignissen und diese selbst bestimmen: Wie wäre das?

Um ein Spiel selbst zu erstellen, muss man vom Programmieren anfangs eigentlich noch gar nichts verstehen. Denn zuallererst braucht man eine Idee und dann einen Plan.

Wovon soll das Spiel handeln? Welche Geschichte soll es erzählen? Personen, Orte und Ereignisse, all das führt zu einem Plan, der umfasst, was zu diesem Spiel gehören soll. Und erst, wenn der Plan »steht«, kann die eigentliche Umsetzung in ein Programmprojekt beginnen. Dann allerdings sollte man schon möglichst gut programmieren können.

Wie du auf eine gute Idee kommst, wie du deine eigene Geschichte »strickst« und wie du einen Plan für ein Spiel aufstellst, wirst du in diesem Buch nicht erfahren. Hier bekommst du einen Kasten voller Werkzeuge, mit dem du deinen Plan in ein Spiel umsetzen kannst.

Wir wollen auch gar nicht so hoch hinaus: Ein professionelles Game wird heutzutage ja von einer ganzen Gruppe von Leuten erstellt, darunter Designer, Künstler, Techniker und nicht zuletzt natürlich Programmierer.

Trotzdem dauert die Arbeitszeit häufig mindestens Monate, wenn nicht Jahre. Die Beteiligten machen einen Vollzeitjob, es ist ihr Beruf. Hier hast du als Einzelgänger nur eine Chance, wenn deine Spiel-Idee so hervorragend und einmalig ist, dass sie andere überschattet.

Bleiben wir also auf dem harten Boden der Tatsachen und planen nicht ein gigantisches Meisterwerk, sondern kümmern uns um solide Grundlagen. Wenn du die beherrschst, hast du durchaus Voraussetzungen, auch einmal an einem professionellen Spielprojekt mitzuwirken.

WELCHE WERKZEUGE BENÖTIGEN WIR?

Um Spiele im 2D- und 3D-Bereich zu erstellen, brauchen wir als Herzstück eine sogenannte *Game-Engine*. Sie muss mit physikalischen Gesetzen umgehen können, damit die Spielwelt mit ihren Figuren und Ereignissen möglichst echt wirkt. Und sie muss komplexe grafische Effekte beherrschen, damit das Ganze auch optisch etwas hermacht.

Das brauchst du:

- ◇ Mit **Unity** (früher Unity3D) haben wir nicht nur eine vollwertige Engine, die so vielfältige Möglichkeiten bietet, dass dieses Buch allein nicht reicht, um alles zu beschreiben. Mit dem visuellen Editor lässt sich ein Spiel bequem erstellen – und das nicht nur für Windows, sondern auch für andere Plattformen, wie z.B. Android oder iOS.

Wir verwenden hier die derzeit aktuelle Unity-Version 2020 (für eventuelle Neuerungen schaust du am besten unter *unity.com* nach).

Die visuelle Entwicklungsumgebung, die Unity mit sich bringt, bietet schon viele Möglichkeiten, sich Elemente für ein Spiel zusammenzustellen. Damit aber alles wie gewünscht funktioniert und um die volle Leistung der Engine abzurufen, braucht man ein Programmiersystem:

- ◇ **Visual Studio** bietet die Möglichkeit, die für Unity nötigen Skripts in einer passenden Programmiersprache, nämlich C#, zu erstellen. Dieses System gibt es kostenlos bei Microsoft (unter *visualstudio.microsoft.com*). Es kann aber auch über die Unity-Installation automatisch so eingerichtet werden, dass es aus Unity heraus einsetzbar ist.

Damit wir auch eigene Welten und eigene Akteure für unsere Spiele erstellen können, gibt es in Unity weitere leistungsstarke Hilfsmittel, wie z.B. einen Editor u.a. für Landschaften und Bäume. Viele mit anderen Programmen hergestellte Figuren und Elemente lassen sich leicht in Unity importieren und dort einsetzen.

WAS BIETET DIESES BUCH?

Richtig los geht es wie bei jedem Buch in Kapitel 1. Bist du ein erfahrener C#-Programmierer, so wird es für dich kein Problem sein, auch dort zu starten. Willst du aber dein C#-Wissen lieber noch mal überprüfen oder festigen, dann kannst du dies in einem vorgeschalteten Kapitel tun, das du zum Download auf der Homepage www.mitp.de/0170 findest.

Ansonsten erfährst du hier u.a.,

- ⊙ wie man in Unity Spiele mit Sprites programmieren kann,
- ⊙ wie man den 1st-Person und den 3rd-Person-Modus einsetzt,
- ⊙ wie man Landschaften gestaltet und einfache Gebäude baut,
- ⊙ wie man klettert, schwimmt und taucht,
- ⊙ wie man einfache Figuren erstellt und sie animiert,
- ⊙ wie man künstliche Intelligenz nutzen kann,
- ⊙ wie man Effekte erzeugt und Sound einsetzt.

Im **Anhang** gibt es dann noch zusätzliche Informationen, z.B. wie man Unity und Visual Studio installiert und wie man Fehler vermeidet.

WIE ARBEITE ICH MIT DIESEM BUCH?

Um dir den Weg vom ersten 2D-Projekt bis zu einem 3D-Game mit Akteuren einfacher zu machen, gibt es einige zusätzliche Symbole, die ich dir hier gern erklären möchte:

ARBEITSSCHRITTE

- Wenn du dieses Zeichen siehst, heißt das: Es gibt etwas zu tun. Damit kommen wir beim Programmieren Schritt für Schritt einem neuen Ziel immer näher.

Grundsätzlich lernt man besser, wenn man Objekte selbst einsetzt oder einen Programmtext selbst eintippt oder ändert. Aber nicht immer hat man große Lust dazu. Weil alle Projekte im Buch auch in einem Download-Paket verfügbar sind, findest du hinter einem Programmierschritt auch den Namen des jeweiligen Projekt-Ordners (z.B. GAME1). Wenn du also bestimmte Projekte nicht selbst erstellen willst, kannst du stattdessen die dazugehörigen Dateien herunterladen (zu finden unter mitp.de/0170).

FRAGEN UND AUFGABEN

Am Ende eines Kapitels gibt es jeweils eine Reihe von Fragen und Aufgaben. Diese Übungen sind nicht immer ganz einfach, aber sie helfen dir, deine Spiele noch besser zu entwickeln. Lösungen zu den Aufgaben findest du auch im Download von der mitp-Seite (dort findest du auch die Programme zu den Aufgaben).

Du kannst sie dir alle im Editor von Windows oder auch in deinem Textverarbeitungsprogramm anschauen. Oder du lässt sie dir ausdrucken und hast sie dann schwarz auf weiß, um sie neben deinen PC zu legen.

NOTFÄLLE



Vielleicht hast du irgendetwas falsch gemacht oder etwas vergessen. Oder es wird gerade knifflig. Dann fragst du dich, was du nun tun sollst. Bei diesem Symbol findest du eine Lösungsmöglichkeit. Notfalls kannst du aber auch ganz hinten im Anhang B nachschauen, um ein paar Hinweise zur Pannenhilfe zu finden.

WICHTIGE STELLEN IM BUCH



Hin und wieder findest du ein solch dickes Ausrufezeichen im Buch. Dann ist das eine Stelle, an der etwas besonders Wichtiges steht.



Wenn du ein solches »Wow« siehst, geht es um ausführlichere Informationen zu einem Thema.

WAS BRAUCHST DU FÜR DIESES BUCH?

Du findest das Unity-Paket als komplette Entwicklungsumgebung zu Download und Installation auf der entsprechenden Seite von **Unity**:

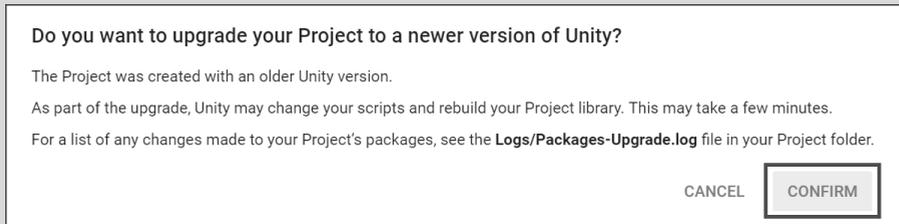
<https://unity.com/>

Zusätzlich gibt es dort noch eine Sammlung von Hilfsmitteln, die sogenannten Assets. Du findest dort auch eine Menge Zusatzmaterial. Während Unity dich nichts kostet, solange du nicht damit Geld verdienen willst, sind viele Zusätze nicht kostenlos.

Die Beispielprojekte in diesem Buch findest du ebenso wie die Lösungen zu den Aufgaben auf der Homepage des Verlags in der gerade aktuellen Version:

<http://www.mitp.de/0170>

Die Programmbeispiele sind mit einer Unity-Version erstellt, die vielleicht schon nicht mehr die allerneueste ist, wenn du dieses Buch kaufst oder eine Weile besitzt. Denn Unity wird natürlich ständig weiterentwickelt. Wenn du eines der Projekte in einer neueren Unity-Version öffnest, bekommst du eine solche Meldung:



Mit einem Klick auf CONFIRM wird das alte Projekt für die aktuelle Unity-Version umgewandelt.

BETRIEBSSYSTEM

Die meisten Computer arbeiten heute mit dem Betriebssystem Windows. Am besten geeignet ist Version 10. Mit Unity lassen sich allerdings nicht nur Spiele für Windows entwickeln, sondern auch für andere Systeme wie Googles Android, Apples iOS, ja sogar Microsofts Xbox und Sonys Playstation.

SPEICHERMEDIEN

Auf jeden Fall benötigst du etwas wie einen USB-Stick oder eine SD-Card, auch wenn du deine Programme auf die Festplatte speichern willst. Auf einem externen Speicher sind deine Arbeiten auf jeden Fall zusätzlich sicher aufgehoben.

Gegebenenfalls bitte deine Eltern oder Lehrer um Hilfe.

WIE GUT KENNST DU C#?

Vielleicht kennst du bereits eine andere Programmiersprache, wie z.B. JavaScript, Python oder auch Basic. Dann sollte dir der Umstieg auf C# nicht schwer fallen. Im Idealfall hast du bereits in C# programmiert und bist mit den Grundlagen vertraut.

Aber auch wenn du keine Ahnung vom Programmieren, aber unbändige Lust auf die Spiele-Programmierung mit Unity hast, kannst du einfach so in Kapitel 1 einsteigen und versuchen, mitzukommen. Programmiert wird erst ab Kapitel 2.

HINWEISE FÜR LEHRER

Dieses Buch lässt sich selbstverständlich auch für den Informatik-Unterricht verwenden. Dort setzt natürlich jeder Lehrer seine eigenen Schwerpunkte. Aber wenn es z.B. um eine Programmier-AG oder einen Informatikkurs mit Schwerpunkt Spieleprogrammierung geht, lässt sich dieses Buch in Ergänzung zu Ihrem Lehrmaterial gut einsetzen. Mit Unity steht Ihnen und Ihren Schülern ein mächtiges Entwicklungswerkzeug zur Verfügung.

Was die Programmierung angeht, so wird hier die Sprache C# genutzt, die der Programmiersprache Java recht ähnlich, aus meiner Sicht aber einfacher zu erlernen und einzusetzen ist, weil C# die Vorteile vieler anderer Sprachen vereint und nur wenige Nachteile hat.

Die einzelnen Spielprojekte werden vorwiegend über den visuellen Editor von Unity erstellt, die Programmierung in C# dient der Erstellung von Scripts, um den Spielverlauf zu steuern. Zahlreiche Fragen und Aufgaben mit Lösungen helfen, Gelerntes zu festigen und zu vertiefen.

ÜBUNGSMEDIEN

Für den Informatik-Unterricht sollte jeder Schüler ein eigenes externes Speichermedium haben, um darauf seine Versuche zu sichern. So wird verhindert, dass sich auf der Festplatte des Schulcomputers mit der Zeit allerlei »Datenmüll« ansammelt. Außerdem dient der eigene Datenträger dem Datenschutz: Nur der betreffende Schüler kann seine Daten manipulieren.

REGELMÄßIG SICHERN

Es kann nicht schaden, die Programmdateien, an denen gerade gearbeitet wird, etwa alle zehn Minuten zu speichern. Denn Computer pflegen gern gerade dann »abzustürzen«, wenn man seine Arbeit längere Zeit nicht gespeichert hat.

1 **DAS ERSTE PROJEKT**

Du möchtest natürlich gleich hier an Ort und Stelle dein erstes Spiel erstellen. Daraus wird nichts. Noch nicht. Als Erstes brauchst du Geduld. Und Ausdauer. Hier starten wir aber schon die Maschine, mit der wir später einiges zaubern werden, das sich sehen und spielen lässt. Wir fangen mit einem Projekt an. Und wir spielen auch schon mal ein bisschen mit einem Objekt herum.

In diesem Kapitel lernst du

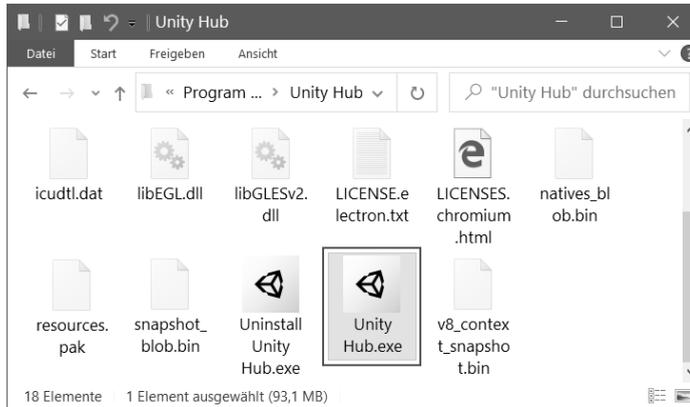
- ⊙ wie man Unity startet
- ⊙ deine »Werkbank« kennen
- ⊙ wie man Spiel-Objekte in einem Projekt einsetzt
- ⊙ ein bisschen über Spiele-Physik
- ⊙ etwas über 2D und 3D
- ⊙ wie man die Position und Größe von Objekten ändert
- ⊙ wie man Unity beendet

UNITY STARTEN

Bevor wir mit dem Basteln anfangen können, muss das Game-Entwicklungssystem Unity installiert werden. Wie das geht, erfährst du in **Anhang A**. Wenn du dir das Einrichten nicht allein zutraust, solltest du dir von jemandem helfen lassen.

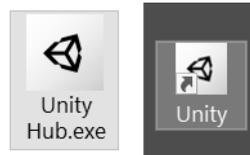
Eine Möglichkeit, Unity zu starten, ist diese:

- Öffne den Ordner, in den du Unity untergebracht hast (bei mir ist das der Unterordner UNITY HUB im Ordner PROGRAMME auf Laufwerk C:).



Hier suchst du unter den vielen Symbolen eines von denen heraus, die wie eine Art schwarzer Würfel aussehen, und zwar das mit dem Namen UNITY HUB.EXE.

- Starte das Programm mit einem Doppelklick auf das Symbol.



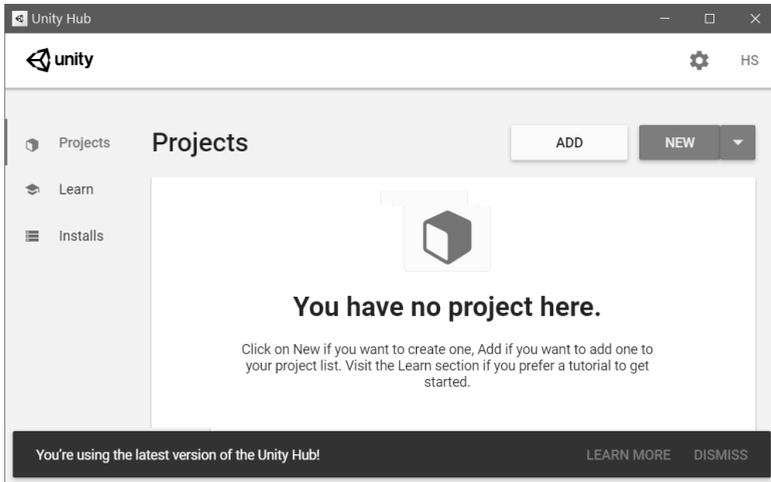
Ich empfehle dir, eine **Verknüpfung** auf dem Desktop anzulegen:

- ❖ Dazu klickst du mit der rechten Maustaste auf das entsprechende Unity-Symbol (Unity Hub.exe). Im Kontextmenü wählst du **KOPIEREN**.
- ❖ Dann klicke auf eine freie Stelle auf dem Desktop, ebenfalls mit der rechten Maustaste. Im Kontextmenü wählst du **VERKNÜPFUNG EINFÜGEN**.
- ❖ Es ist sinnvoll, für das neue Symbol auf dem Desktop den Text `unity hub.exe - Verknüpfung einfach durch Unity` zu ersetzen.

Von nun an kannst du auf das neue Symbol doppelklicken und damit Unity starten.



Je nach Computer kann es eine Weile dauern, bis Unity geladen ist. Einige Zeit später landest du in einem neuen Fenster.

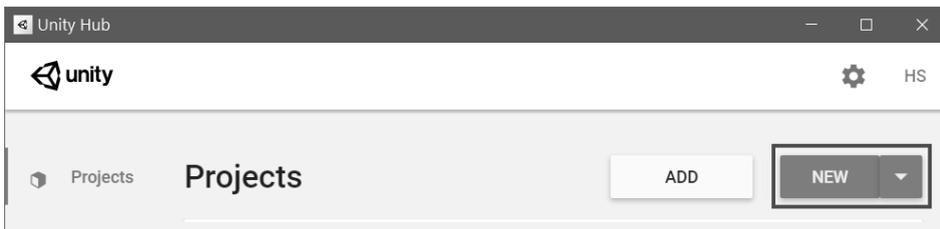


Startest du Unity zum allerersten Mal seit der Installation, dann musst du es wahrscheinlich noch aktivieren. Wenn du das nicht allein hinkriegst, schau mal in Anhang A nach.



Dort hast du nun die Möglichkeit, ein Projekt zu öffnen (wenn vorhanden). Oder eines neu zu erstellen.

➤ Klicke auf NEW.

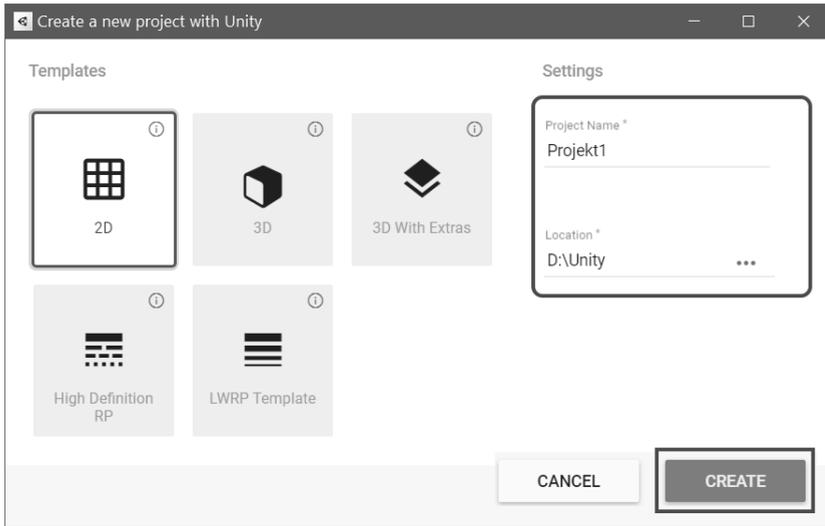


➤ Im neuen Dialogfeld wählst du die Einstellung 2D. Mit 3D beschäftigen wir uns später.

➤ Gib im Feld für PROJECT NAME den Namen des neuen Projekts ein. Und dann bei LOCATION den Ordner, wo du dein Projekt unterbringen willst. (Wenn du alles so stehen lässt, schafft sich Unity seinen eigenen Ordner für deine Spiel-Projekte.)

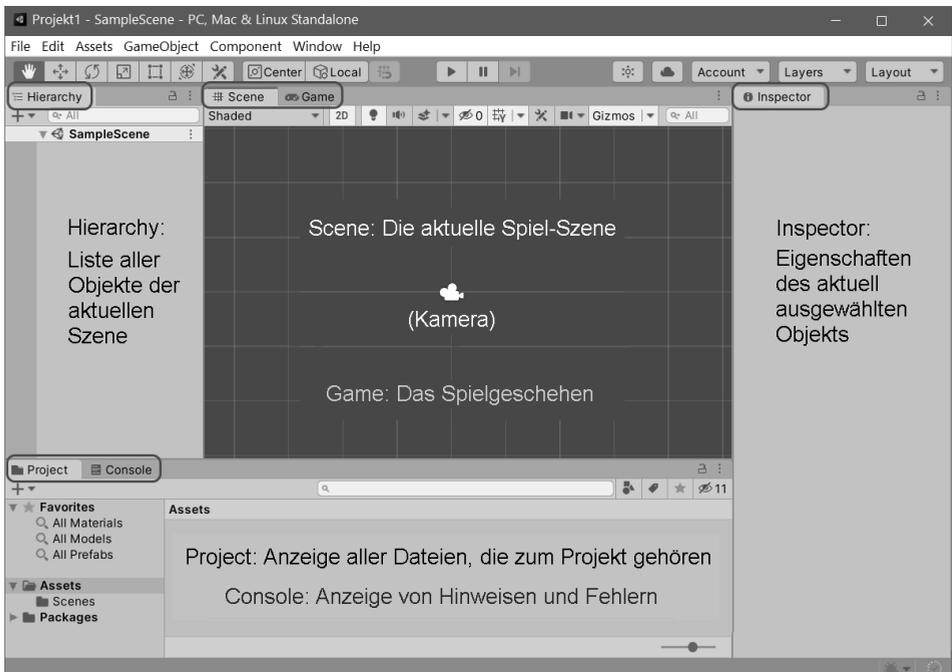
Ich benutze einen Ordner UNITY und nenne mein erstes Projekt schlicht und einfach PROJEKT1.





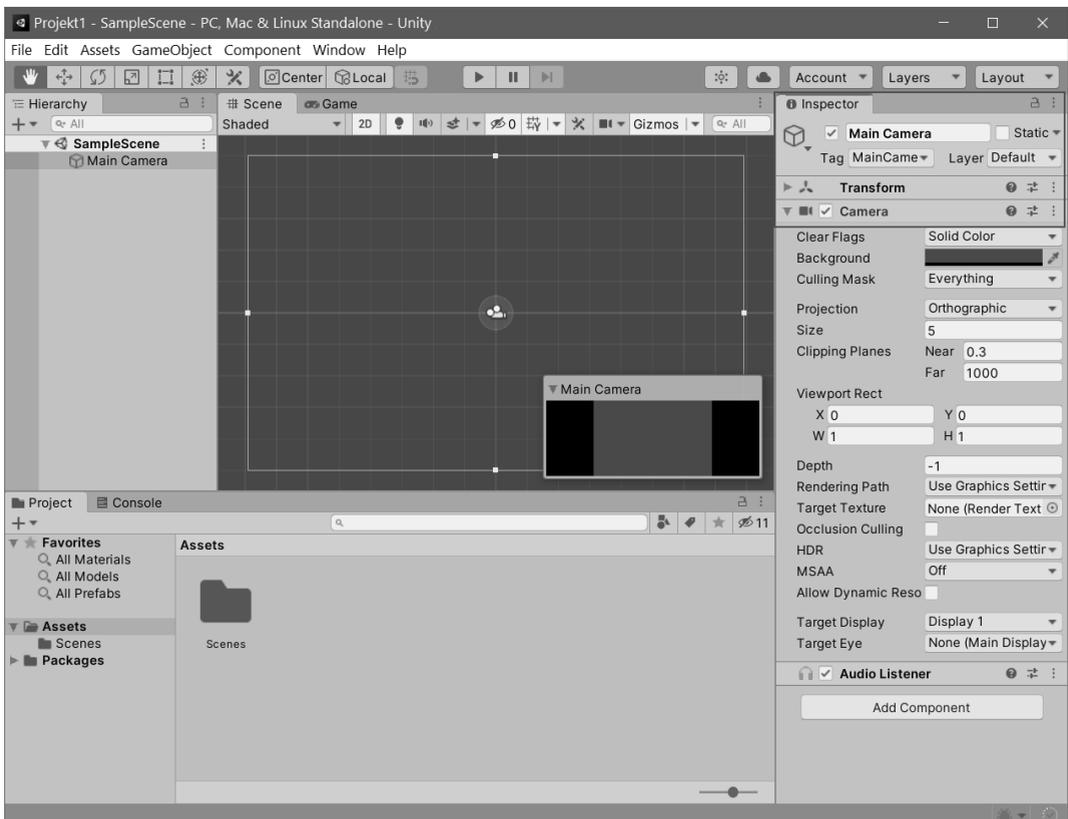
➤ Klicke dann auf CREATE.

Und endlich zeigt uns Unity sein wahres Gesicht (was eine ganze Weile dauern kann). Schauen wir uns erst einmal die Aufteilung der vier wichtigsten Fensterbereiche an:



◇ Im SCENE-Fenster sieht man erst einmal nur die Kamera. Für ein Spiel brauchen wir dann noch mindestens ein weiteres Objekt wie eine Kugel oder eine Figur.

- ◇ Dahinter liegt das GAME-Fenster: Hier siehst du dein Spiel in Echtzeit ablaufen, wenn du es durch einen der darüberliegenden Buttons gestartet hast.
- ◇ Im HIERARCHY-Fenster ist bis jetzt nur die MAIN CAMERA aufgelistet. Dort stehen dann später alle Objekte, die zur Szene eines Spiels gehören (jedes Spiel könnte also mehrere Szenen haben).
- ◇ Das PROJECT-Fenster erfasst die Ordner mit dem gesamten Zubehör für das ganze Spiel. Dazu gehören natürlich u.a. auch Programmteile. Bilder, die du als Spielobjekt einsetzen willst (wie z.B. eine Kugel oder eine Figur) lassen sich einfach mit der Maus aus einem Ordnerfenster unter Windows hier hineinziehen. Womit die entsprechende Datei ins Projekt kopiert wird.
- ◇ Dahinter findest du das CONSOLE-Fenster, das sich bei Fehlern meldet. Außerdem lassen sich dort Werte, z.B. von Spiel-Objekten, anzeigen.
- ◇ Um sich die Eigenschaften eines Objekts nicht nur anzuschauen, sondern auch bearbeiten zu können, gibt es das INSPECTOR-Fenster.
- Klicke mal im HIERARCHY-Fenster auf MAIN CAMERA. Dann verändert sich nicht nur die Darstellung im SCENE-Fenster, sondern der INSPECTOR zeigt plötzlich eine ganze Menge an (ändern solltest du allerdings daran nichts).



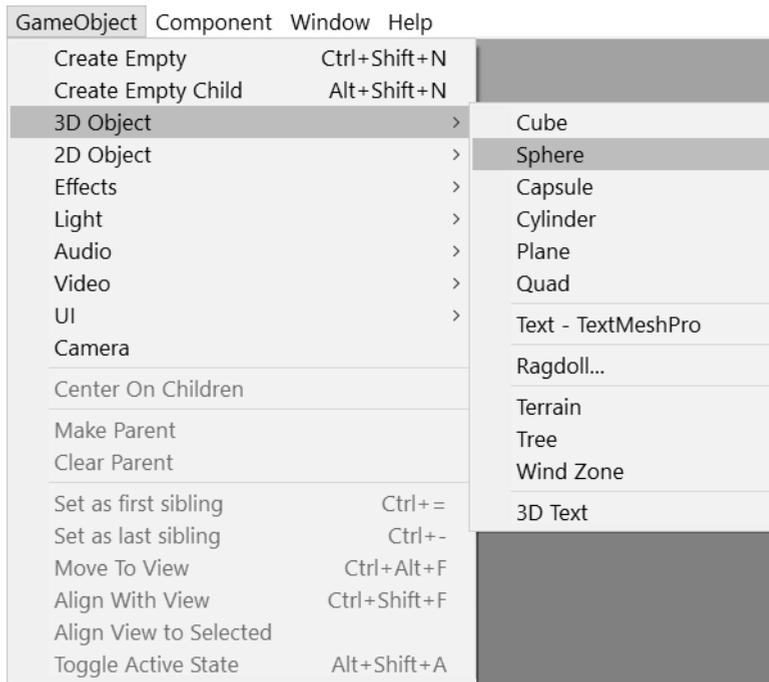
Um das Hauptmenü und die darunterliegenden Symbole geht es erst mal nicht, damit bekommen wir später noch genug zu tun.



EIN OBJEKT ZUM SPIELEN

Wir beginnen mit etwas Einfachem. Wir brauchen eine Kugel und die soll sich über das Spielfeld bewegen lassen. Mit der Maus oder mit den Tasten zum Beispiel.

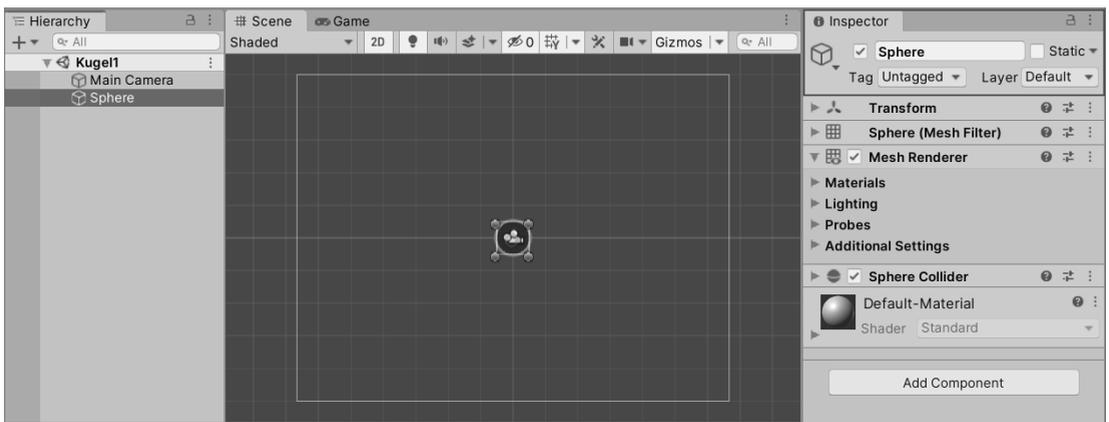
- Kommen wir zum Hauptmenü. Klicke dort auf GAMEOBJECT und dann auf den Eintrag 3D OBJECT. Im Zusatzmenü bekommst du nun eine Auswahl. Klicke auf den Eintrag SPHERE (= Kugel).



Alternativ zum Hauptmenü lässt sich ein solches Objekt auch direkt im HIERARCHY-Fenster erstellen. Dazu öffnest du mit der rechten Maustaste ein Kontextmenü, und über 3D OBJECT ein weiteres. Dort klickst du auf das gewünschte Objekt, wie z.B. SPHERE.



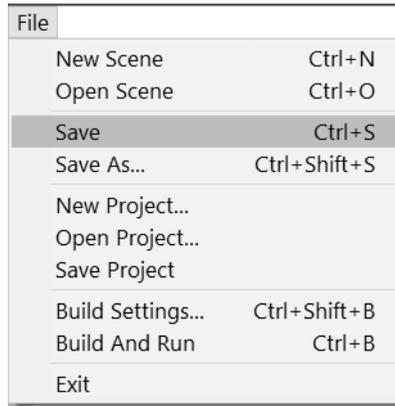
Anschließend taucht im SCENE-Fenster etwas auf, das wie ein Kreis aussieht. Außerdem zeigt der INSPECTOR zahlreiche Informationen über unser neues Spiel-Objekt.



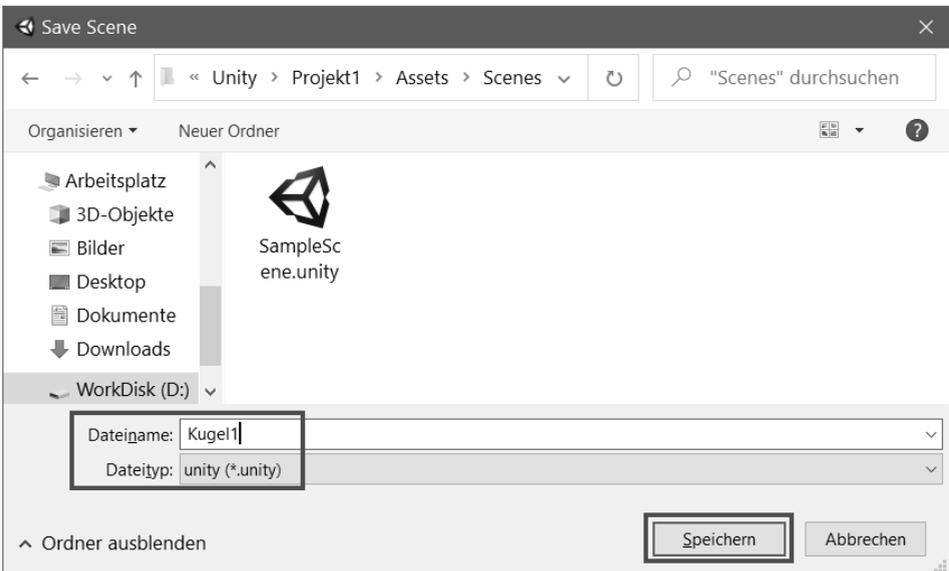
Dargestellt sind nun zwei Objekte: Die Kamera (auf die wir noch zu sprechen kommen) und darunter oder dahinter die von uns erzeugte Kugel.

Nun ist es an der Zeit, die ganze Szene schon einmal zu speichern.

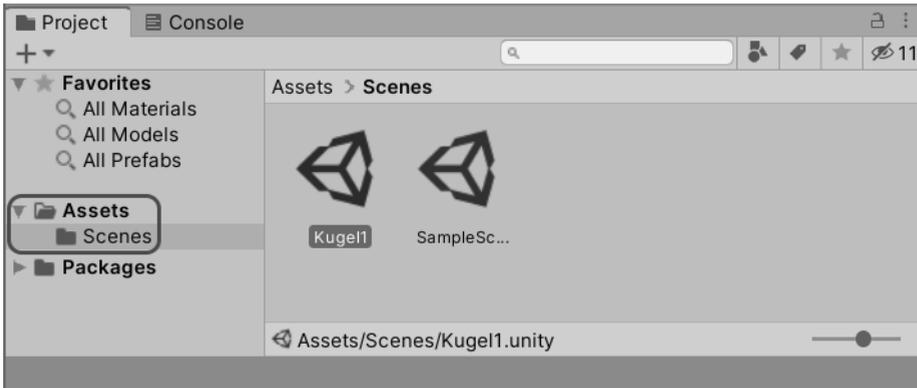
➤ Klicke auf FILE und SAVE.



Bei Unity heißt diese Szene zuerst SAMPLESCENE. Passt dir der Name nicht, dann klicke auf **SPEICHERN UNTER** und gib im Dialogfeld einen Namen ein, z.B. Kugel1 (wenn dir nichts Besseres einfällt). Die Kennung UNITY wird automatisch angefügt. Dann klicke auf **SPEICHERN**.

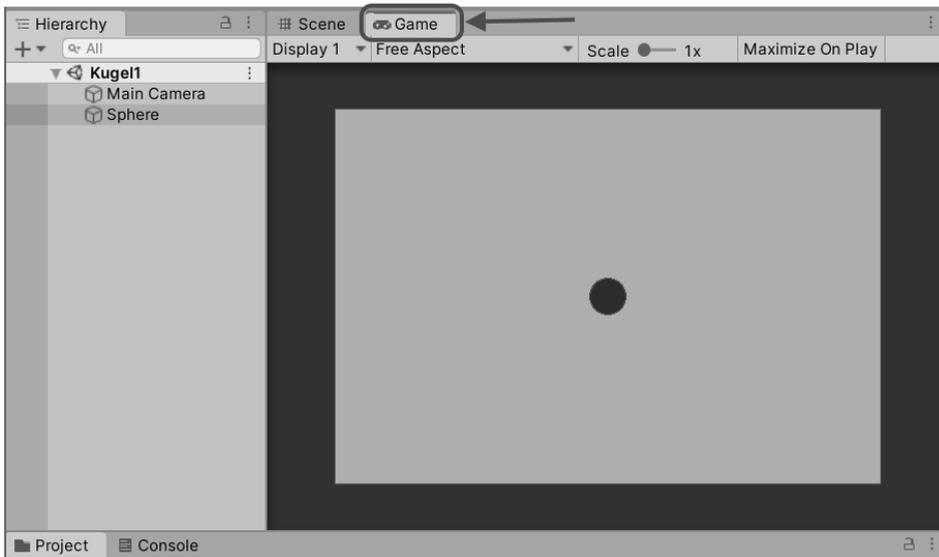


Wenn du anschließend im PROJECT-Fenster auf **ASSETS** klickst, siehst du da den Ordner **SCENES**, darin befindet sich das Symbol für die Szenen-Dateien.



Und nun schauen wir uns diese Szene einmal genauer an, und zwar in einem anderen Fenster.

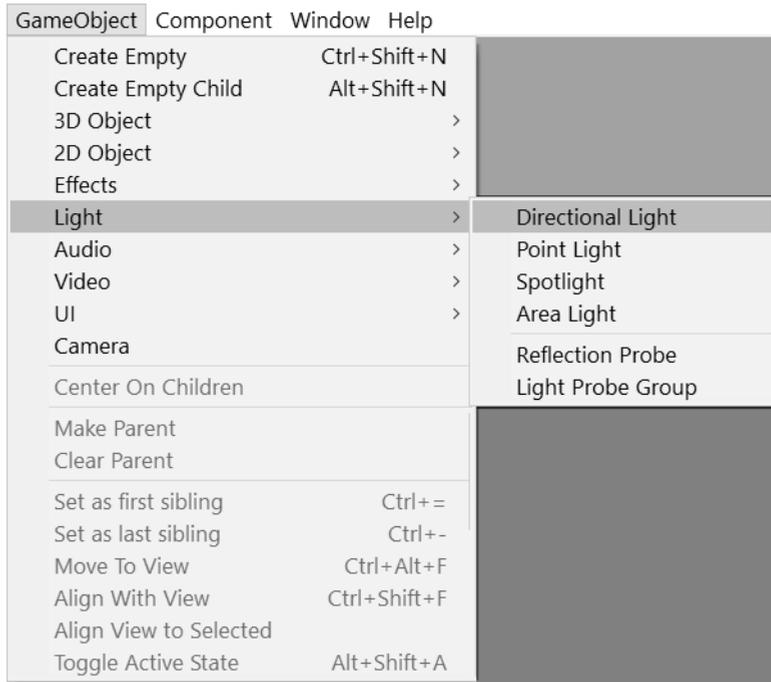
➤ Dazu klickst du auf den Reiter mit dem Text GAME (direkt rechts neben dem SCENE-Reiter).



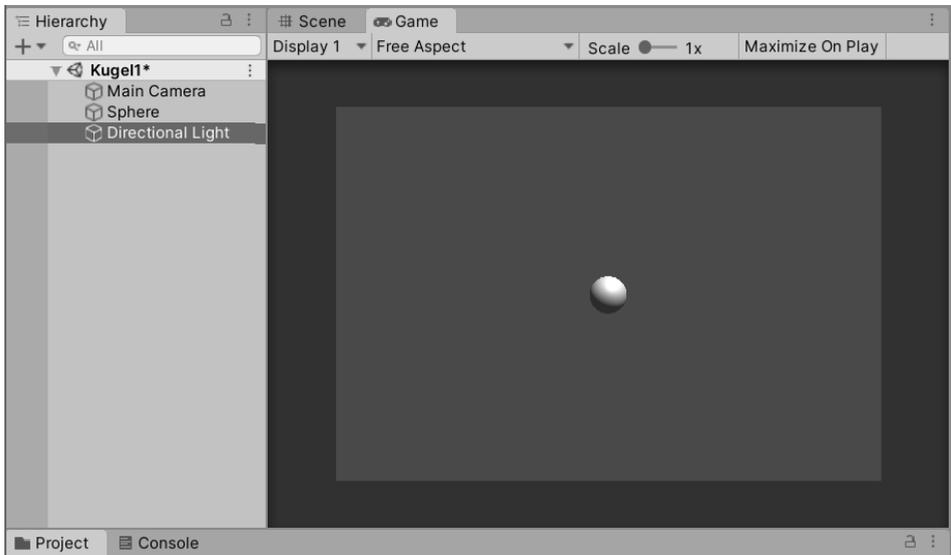
Du landest im GAME-Fenster. Das hat sich bis jetzt hinter dem SCENE-Fenster versteckt.

Hier kannst du sehen, wie dein Spiel in Aktion aussieht. Bis jetzt ist da noch alles dunkel und es bewegt sich nichts. An der Dunkelheit können wir sofort etwas ändern.

➤ Klicke im Hauptmenü auf GAMEOBJECT und dann auf LIGHT. Im Zusatzmenü klickst du auf den Eintrag DIRECTIONAL LIGHT.

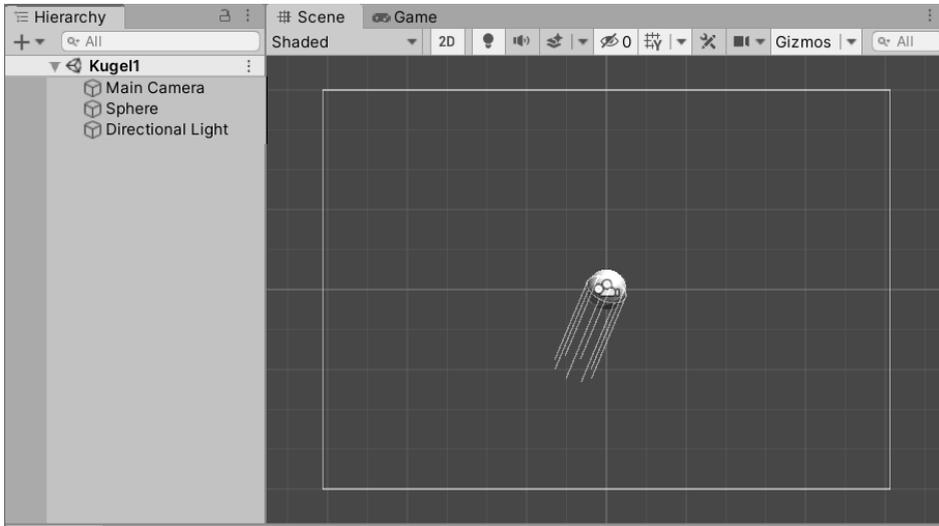


Kurz darauf siehst du deine Kugel in einem anderen Licht. Nur im GAME-Fenster?



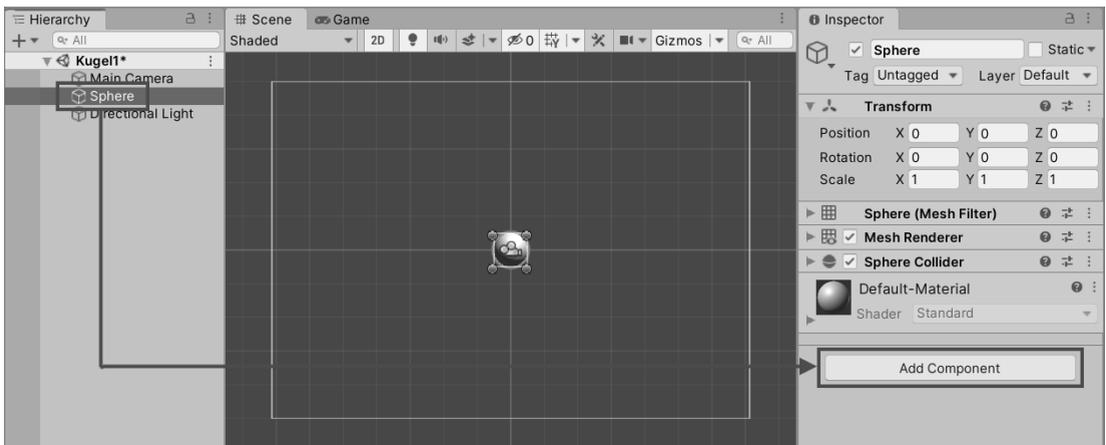
➤ Klick dich doch mal zurück zum SCENE-Fenster.

Dort gibt es auch eine Änderung, wie du sehen kannst. Das neue Spiel-Objekt wird als Symbol aus Strahlen dargestellt.



Da haben wir nun eine Kugel, die sich sonnt. Aber sie soll sich bewegen. Doch wie kriegen wir sie dazu? Zuerst einmal sollten wir diesem »Ding« physikalische Eigenschaften geben. Dass das Objekt aussieht wie eine Kugel, heißt noch nicht, dass es sich auch wie eine Kugel aus einem bestimmten Material verhält.

➤ Markiere also im HIERARCHY-Fenster (links) den Eintrag SPHERE. Dann suche im INSPECTOR-Fenster (rechts) nach einem Button mit der Aufschrift ADD COMPONENT und klicke darauf.



STICHWORTVERZEICHNIS

// 60

1st Person 136
2D-3D 39
2D-Ansicht 41, 365
3D-Ansicht 41
3rd Person 136

A

Absolutwert 332
Account 154, 400
AddForce 56, 212
AI 290
Aktivierung 19
Alpha-Wert 262
Anchor 368
Android 384
Angle 364
Animation
 Attacking 326
 Clip 279, 312
 Datei 313
 Default 316
 Dying 314
 erzeugen 281
 Exit Time 320
 Fenster 312
 Keyframe 282
 Moving 283
 Ordner 277
 Position 280
 Rotation 280
 Speed 322
 Trigger 287
 Zwischenwerte 282
Animator 279
 Condition 319
 Controller 279, 315
 enabled 288
 Fenster 316
 Parameter 318

 speed 288
 State 316
 Transition 319
AnimatorTrigger 288
Application
 Quit 388
Area Light 239
Asset 48
 exportieren 69
 importieren 71
 Package 152
Asset Store 156, 238
Assets 14
 importieren 172
attachedRigidbody 213
Attack 338
AttackControl 338
Audio
 Clip 353
 Formate 352
 Hintergrund 358
 Play 355
 PlaySound 354
 Stop 354
Audio Listener 352
AudioSource 352
Avatar 279
Axes 106

B

back 59
Bauelement 220
Baum
 entfernen 181
 ersetzen 181
 hinzufügen 180
 Kollision 182
Baum Einstellungen 181
Baumaterial 217
Baum-Werkzeug 179

Behaviour 50
 Bend Factor 200
 Benutzerschnittstelle 365
 Billboard 185
 Blender 171, 238
 Boden
 Ausrichtung 295
 Box Collider 196
 Button
 Negative 106
 Positive 106

C

C# 50
 CameraSwitch 150
 Canvas 366
 Capsule 144
 Capsule Collider 196
 CharacterController 88, 140
 Chaser 338
 Child 146, 366
 class 55
 ClimbControl 249
 Climber 249
 Cluster 184
 Collider 35, 36, 99, 196
 anpassen 260
 Collider Trigger 124
 CollisionControl 212, 310
 Color 97, 126
 Component 27
 Conditions 319
 Cone 347
 Console-Fenster 21
 Container 224, 236
 Cos 363
 Create New Clip 314
 CreatureAudio 357
 CreatureControl 294
 CreatureHealth 369, 375

D

Debug 144, 405
 Deklaration 92
 deltaTime 64, 355
 Destroy 289, 311

Detail-Werkzeug 183
 Dezimal-f 91
 Directional Light 134, 239
 down 59
 DrawLine 405
 DrawRay 405
 Drehung
 Ausgangspunkt 362
 Drop Height 380
 Dummy 144, 274
 Du-Perspektive 136

E

Editor 51, 52
 Ellipsoid 275
 else 114, 253
 Eltern-Objekt 366
 Emission 347
 Empty Object 196, 224, 259
 Empty Trigger 248
 enabled 150
 Energie
 Anzeige 372
 Kontrolle 371
 Verlust 373
 wiederherstellen 381
 Engine 12
 Er/Sie-Perspektive 136
 Ereignis-Methode 126, 213
 eulerAngles 363
 EventSystem 366
 Export 69

F

false 150
 Farbe 97
 f-Dezimal 91
 Find (Objektnamen) 250, 267
 First Person 136
 FixedUpdate 64
 Flatten 257
 float 91
 Fog 261
 fogColor 262
 fogDensity 262
 forward 59

Frame 64
Freeze 123
Fullscreen 386

G

Game
 Fenster 25
 Play 31
Game Build 383
Game Over 376
Game-Engine 12
Game-Fenster 21
GameObject 22, 79
Gamepad 107
Game-View maximieren 178
Gegner
 erstellen 274
Geräusche 352
Gesundheit 364
GetAxis 104
 Maus 142, 194
 Tasten 106
GetButton 108
GetComponent 56, 92
GetKey 57
GetKeyUp 256
GetTouch 107
Gizmo 149
Gras
 Einstellungen 184
Gravitation 31
Grenze
 unsichtbar 197
GUI
 Canvas 366
 Element 365
 Image 365
 Text 370

H

Halbdrehung 256
Hand-Symbol 202
Health 369
Height Mesh 296
Hierarchy-Fenster 21
Hintergrundsound 358

I

Ich-Perspektive 136
if 57
Import 71
Input 57, 140
Input Axes 106
Input-Manager 104, 105
Inspector-Fenster 21
Installation 391
int 91, 322
Invoke 330, 339
isClimbing 251
isGrounded 109
isKinematic 214
isPlaying 351

J

Joystick 107
Jump 108
Jump Distance 380

K

Kamera
 bewegen 136
 drehen 139
 springen 140
 umschalten 148
 verbinden 146, 148
 Verknüpfung 151
 verschieben 139
Keyframe 282
Kind-Objekt 366
Kinematik 303, 324
Klammer
 geschweift 55
 rund 55
Klettern 250
Kletter-Trigger 248
Kollision 31, 212
 Aufprall 311
 Bäume 182
 Schaden 310
Kommentar 60
Komponente
 Rigidbody 29
Kontext-Menü 36, 146

Kontrollstruktur 57, 114
 Konvertieren 15, 404
 Koordinatensystem 39
 Korrektur 295
 Kreatur
 Beine 275
 Container 276
 erstellen 274
 Körper 275
 Rigidbody 277
 Künstliche Intelligenz 290, 293

L

Layer 175
 Leertaste 108
 left 59
 Leiter 255
 Library 209
 Licht 239
 Lichteinstellungen 241
 Light 239
 Lighting 235
 Linecast 342
 LineRenderer 340
 localScale 372
 Lock View 211, 248
 Log 144, 405
 LookAt 364
 LTS 398

M

Material 95
 Mathf.Abs 332
 Mesh 170
 Mesh Collider 196
 Mesh Renderer 327
 Methode 55
 MonoBehaviour 55
 MonoDevelop 213
 Mouse X 142
 Mouse Y 194
 Move 93
 moveDirection 213
 MovePosition 63
 MoveVector 107

N

Navigation
 Karte 291
 not walkable 308
 Umfeld 291
 vorbereiten 290
 Navigation-Mesh 380
 NavigationMesh 290
 Navigator
 enabled 297
 stoppen 299
 NavMeshAgent 293
 Nebel 261
 Neues Objekt 22, 31, 79
 Neues Projekt 19

O

Objekt
 Container 224
 deaktivieren 289
 drehen 37
 Ebene 79
 entfernen 289
 Größe 37
 Kugel 22
 leer 196, 224
 Licht 25, 94
 neu 22, 31, 79
 Quader 31
 Standardmaße 161
 umbenennen 43
 verschieben 32
 Zentrieren 211, 248
 Oder-Operator 214
 Official Release 398
 OffMeshLinks 381
 Offset 220
 On 126
 OnCollisionEnter 310
 OnCollisionExit 310
 OnCollisionStay 310
 OnControllerColliderHit 213
 OnTriggerEnter 126, 249, 289
 OnTriggerExit 126, 249, 289
 OnTriggerStay 257
 Opacity 177
 Operator && 214

Operator || 214
Ordner
 erstellen 48, 78
Orthographic 42
other 310

P

Package 152, 153
Package Manager 154, 171
Parameter 318
Parent 146, 366
ParticleSystem 350
Partikel
 Ausstoß starten 351
 Ausstoß stoppen 350
 Bereiche 349
 Color 347
 Emission 347
 Hauptmodul 346
 Shape 347
Partikelsystem 343
 erzeugen 350
Performance 185
Pfeiltaste 55
Pfeiltasten 87, 107
Physics 336
Pinsel-Werkzeug 164
Plane 136
Platformer 119
Plattform 384
Play 351
Player
 Dying 325
PlayerAudio 353
PlayerControl 147, 251, 269
PlayerHealth 369, 371
PlaySound 354
Point Light 239
Position 32
position 63
Prefab 205, 277
 erzeugen 223, 230
 exportieren 232
 importieren 233
private 91
Project Settings 104
Project-Fenster 21

Projekt
 3D neu 134
 Asset-Importe 208
 entschlacken 208
 neu 19
Prototyp 223
public 91

Q

Quaternion.Euler 325
Quelltext 54
Quit 388

R

Random 376
Range 376
Raycast
 Direction 337
 Distance 337
 Position 336
 sichtbar 340
RaycastHit 337
Raycasting 336
Rechenoperator 58
Rect Transform 367
Refresh 381
remainingDistance 298
Renderer
 Color 126
 Sprite 111
Rendern 111, 129, 261
RenderSettings 262
return 117, 214, 325
right 59
Rigidbody 29, 56, 137, 209
 Eigenschaften 212
Rotate 139
Rotation 37, 192
Rückgängig 167, 185

S

Scale 33, 37, 137
Scene
 speichern 23, 80
 Zentrieren 211
Scene-Fenster 20

Schlüssel-Nummer 322, 329
 Schlüsselwort 55
 Schwimm-Modus 268
 Screen 371
 Script 47
 doppelt 379
 Sprache 50
 Script erstellen 49, 84
 Scripteditor 55
 Second Person 136
 SendMessage 339, 356
 SetActive 289
 SetBool 322
 SetDestination 295
 SetPosition 342
 Setup 391
 SetZero 268
 Shader 129, 341
 Shape 347
 Shuriken 343
 Sin 363
 Skalierung 37
 Skybox 367
 Slope Limit 169, 243
 Sound 352
 Sphere Collider 196
 SphereCast 342
 SphereControl 54
 Spiel
 erzeugen 383
 Plattform 384
 Spot Light 239
 Sprite 80
 Sprite Renderer 111
 SpriteRenderer 111
 sqrMagnitude 312
 Standard Assets 156
 Start 55
 Startwerte 91
 State 316
 Machine 316
 Steigung 169, 241
 Step Offset 244
 Stop 350
 stoppingDistance 298
 Strahl
 Dicke 340
 String 91

StringToHash 322
 System
 Collections 54
 Szene
 zoomen 202
 Szene zoomen 166

T

Tauch-Modus 268
 Terrain
 Assets 172
 Baum 180
 Büsche 185
 Collider 196
 erweitern 164
 erzeugen 160
 Gras 184
 Grenzen 195
 Höhe 164
 Höhle 170
 Layer 175
 Maße 163
 Mitte 168
 Pack 171
 Plateau 164, 165
 Size und Opacity 166
 Smooth 166
 Stamp 166
 Steine 185
 Textur 174
 Textur hinzufügen 176
 Werkzeug 162
 Wind 187
 Test-Szene 211
 Textur 95
 auftragen 177
 einsetzen 128
 importieren 127, 173
 Paint 174
 Terrain 174
 Tiling 219
 Third Person 136
 Tiling 219, 238
 Time 64, 355
 Touch 107
 Transform 37, 87, 139
 TransformDirection 143, 193

Transitions 319
Translate 87, 139
Tree-Object 179
Treppe 243
Trigger 124
 Klettern 248
 sichtbar 248
 unsichtbar 248, 259
 Wasser 260
true 150

U

Und-Operator 214
Unity 12
 Account 400
 beenden 44
 Hub 391, 396
 installieren 391
 konvertieren 15, 404
 Menüs 20
 starten 18
Unity Links 388
UnityEngine 54
Unterwasser 261
up 59
Update 55
Upgrade 404
Ursprung 39
using 54

V

Variable 90
 Inspector-Fenster 91
Vector3 58, 138, 213
Vektor 59, 107, 138
velocity 312, 356
Verbindungsoperator 57
Vereinbarung 56

Vergleichsoperator 113
Verknüpfungsfehler 379
Verknüpfungsoperator 214
Verneinungsoperator 252
Visual Studio 12, 51, 86, 398

W

Wasser 203
 Höhe ermitteln 267
Wasser-Trigger 260
Waten-Modus 269
Water Object 203
WaterControl 261
Water-Object 257
Werkzeug
 Baum 179
 Detail 183
 Pinsel 164
Wheel Collider 196
Wind Zone 199
Windeinstellungen 187

X

x-Achse 39

Y

y-Achse 39

Z

z-Achse 39
Zahnrad 163
Zeitverzögerung 330
zero 107
Zoom 33, 166, 202
Zuweisung 56
Zuweisungsoperator 92