

Hans-Georg Schumann



3D-Spiele programmieren

mit

Unity



Ganz einfach ohne Vorkenntnisse



Inhaltsverzeichnis

	Einleitung	9
E.1	Spielentwicklung	9
E.2	Und was ist Unity?	9
E.3	Voraussetzungen	10
1	Das erste Projekt	11
1.1	Unity starten	11
1.2	Ein Objekt zum Spielen	16
1.3	Gravitation und Kollision	26
1.4	2D oder 3D?	32
1.5	Ausblick	39
2	Script-Programmierung	43
2.1	Ein Script erstellen	43
2.2	Klassen und Methoden	49
2.3	if-Strukturen	53
2.4	Schubsen oder schieben?	57
2.5	Mal schwerelos, mal »bouncy«	60
2.6	Import und Export	63
2.7	Ausblick	69
3	Spielfigur als Sprite	71
3.1	Ein neues Spielobjekt	71
3.2	Bilder fürs Sprite	73
3.3	Ein Script für die Figur	76
3.4	Character Controller	79
3.5	Material und Textur	85
3.6	Ausblick	90
4	Jump & Run	91
4.1	Steuersystem	91
4.2	Das richtige Bild	99
4.3	Eigene Methoden	104
4.4	Laufen, Springen, Schubsen	107
4.5	Bouncy Ball	110
4.6	Trigger	112

4.7	Texturen	115
4.8	Ausblick	118
5	Sightseeing in 3D	119
5.1	Einfache 3D-Szene	119
5.2	Bewegte Kamera	124
5.3	Springen und Drehen	126
5.4	Player mit Kamera	130
5.5	3rd oder 1st Person?	135
5.6	Fertig-Player aus der Packung?	140
5.7	Ausblick	146
6	Landschaften	147
6.1	Von der Ebene zum Terrain	147
6.2	Ein Gelände gestalten	151
6.3	Rundgang und Asset-Suche	156
6.4	Landschaftspflege	164
6.5	Vegetation	168
6.6	Noch mehr Details?	174
6.7	Ausblick	179
7	Erde, Wasser, Luft	181
7.1	Auf und ab	181
7.2	Grenzkontrollen	184
7.3	Wind	189
7.4	... und Wasser	192
7.5	Entschlackungskur	195
7.6	Kugel mit Rigidbody	196
7.7	Kollision mit Folgen	199
7.8	Ausblick	202
8	Bauwerke	203
8.1	Baumaterial	203
8.2	Platten legen	209
8.3	Prefab-Transport I	216
8.4	Prefab-Transport II	222
8.5	Innenansichten	224
8.6	Steigungen	229
8.7	Ausblick	233

9	Klettern und Schwimmen	235
9.1	Ein Kletter-Trigger	235
9.2	Der Player lernt klettern	238
9.3	Ein kleiner Schubs	242
9.4	See-Landschaft	247
9.5	Unterwasser-Atmosphäre	251
9.6	Waten, Schwimmen, Tauchen	256
9.7	Bewegungskontrolle	261
9.8	Ausblick	264
10	Animation und Navigation	265
10.1	Ein kleines Monster	265
10.2	Animator und Keyframes	269
10.3	Das »Ding« bewegt sich	274
10.4	Trigger-Animation	280
10.5	Ein Navigator für die Kreatur	282
10.6	Verfolgung an/aus	293
10.7	Hindernislauf	296
10.8	Ausblick	299
11	Leben oder Tod	301
11.1	Angriff und Verteidigung	301
11.2	Tödliche Kugeln	306
11.3	Animationen organisieren	309
11.4	Stehen – Gehen – Sterben	315
11.5	Tod des Players?	321
11.6	Die Kreatur wird zum Monster	324
11.7	Ausblick	331
12	Strahlen, Partikel und Sound	333
12.1	Raycasting	333
12.2	Todesstrahlen	339
12.3	Partikelsysteme	343
12.4	Flammenwerfer	352
12.5	Geräusche	353
12.6	Noch mehr Sound?	358
12.7	Ausblick	362
13	Game Tuning	363
13.1	Die Kreatur rüstet auf	363
13.2	Gesundheits-Balken	366
13.3	Energiekontrolle für den Player	373

Inhaltsverzeichnis

13.4	... und für die Kreatur	379
13.5	Game Over	382
13.6	Aufmarsch der Gegner	386
13.7	Play the Game	390
13.8	Ausblick	396
14	Anhang	397
A.1	Unity installieren	397
A.2	Projekte und Links	407
A.3	Debugging	408
A.4	Kurze Checkliste	409
	Stichwortverzeichnis	411

Das erste Projekt

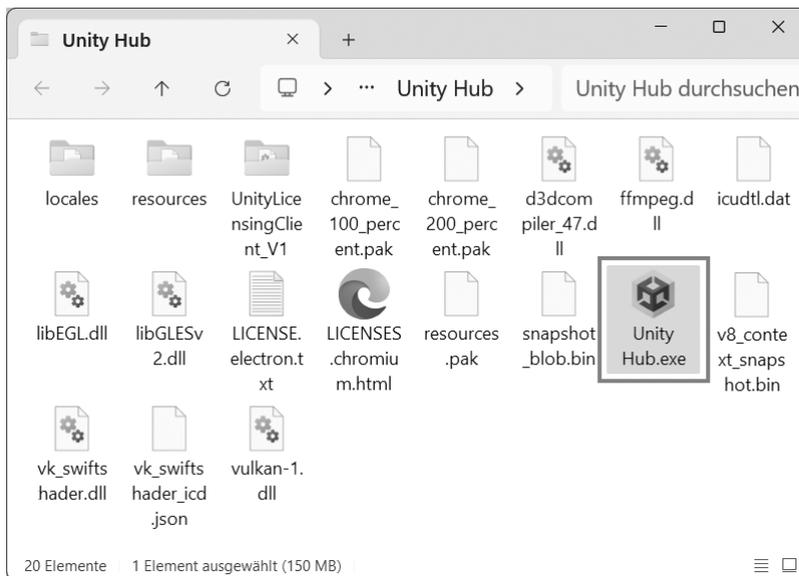
Bevor Sie Ihr erstes Spiel erstellen können, müssen Sie sich noch ein bisschen gedulden. Erst einmal machen Sie sich etwas mit der »Maschine« vertraut, mit der Sie später ein Werk »zaubern« wollen, das sich sehen und spielen lässt. Schon hier beginnen wir mit einem Projekt. Und wir spielen auch schon mal ein bisschen mit einem Objekt herum.

1.1 Unity starten

Bevor wir mit dem »Basteln« anfangen können, muss das Game-Entwicklungssystem *Unity* installiert werden. Wie das geht, steht im *Anhang*. Danach kann es direkt losgehen.

Es gibt mehrere Wege, um Unity zu starten. Einer ist dieser:

1. Öffnen Sie den Ordner, in den Sie Unity installiert haben (bei mir ist das der Unterordner UNITY HUB im Ordner PROGRAMME auf Laufwerk C:).



Kapitel 1

Das erste Projekt

- Suchen Sie nun unter den vielen Symbolen eines heraus, das wie eine Art schwarzer Würfel aussieht, es muss den Namen `Unity Hub.exe` tragen. Dann starten Sie das Programm mit einem Doppelklick auf das Symbol.



Start-Symbol

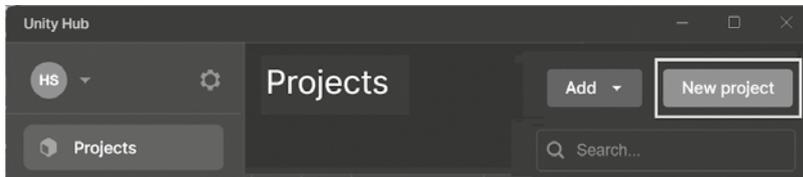
Weil wir Unity ja sehr oft starten werden, empfehle ich hier, eine Verknüpfung auf dem Desktop anzulegen:

- Klicken Sie dazu mit der rechten Maustaste auf das entsprechende Unity-Symbol (`Unity Hub.exe`). Im Kontextmenü wählen Sie **KOPIEREN**.
- Dann klicken Sie auf eine freie Stelle auf dem Desktop, ebenfalls mit der rechten Maustaste. Im Kontextmenü wählen Sie **VERKNÜPFUNG EINFÜGEN**.

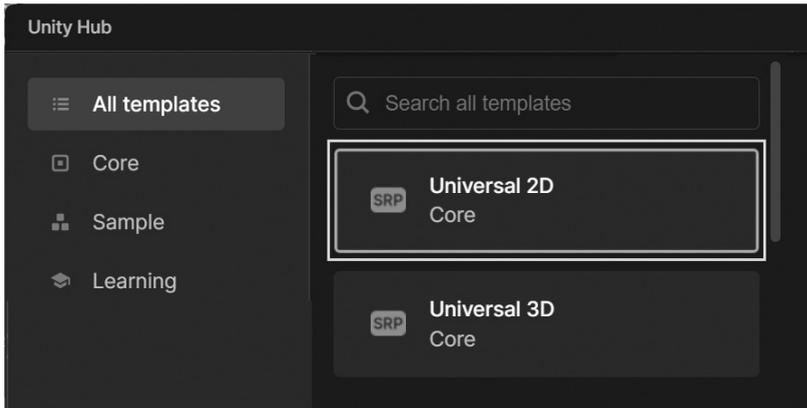
Es ist sinnvoll, das neue Symbol auf dem Desktop umzubenennen, z.B. von `UNITY HUB.EXE – VERKNÜPFUNG` in einfach nur `UNITY`.

Von nun an doppelklicken Sie einfach auf das neue Symbol, und Unity wird gestartet.

Je nach Computer kann es eine Weile dauern, bis Unity Hub geladen ist. Einige Zeit später erscheint ein neues Fenster:

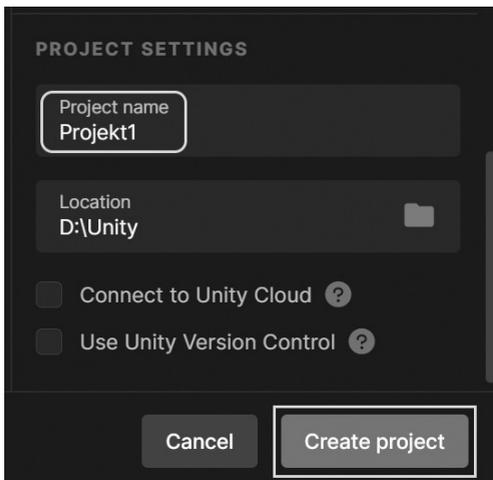


- Klicken Sie dazu auf **NEW PROJECT**.



4. Wählen Sie im neuen Dialogfeld die Einstellung **UNIVERSAL 2D**.
Mit 3D beschäftigen wir uns später noch ausführlich.
5. Geben Sie dann im Feld für **PROJECT NAME** einen Namen für Ihr neues Projekt ein. Bei **LOCATION** sollte der Ordner stehen, in dem das Projekt untergebracht werden soll (wenn Sie nichts eingeben, schafft sich Unity seinen eigenen Ordner für Ihre Spielprojekte.)

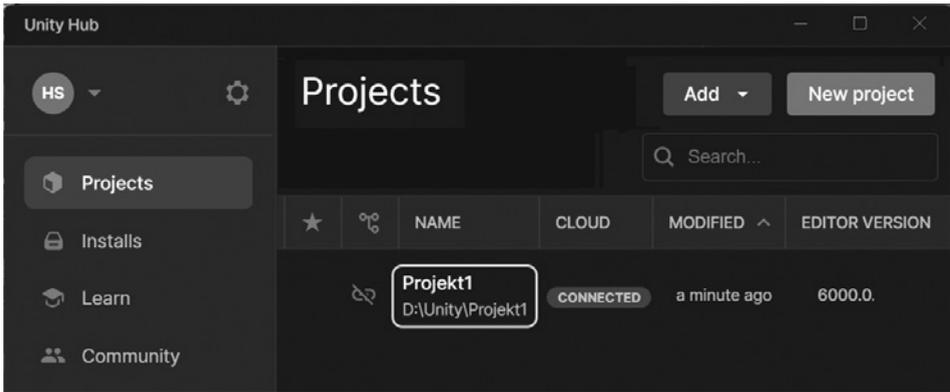
Ich benutze einen Ordner **UNITY** und nenne mein erstes Projekt schlicht und einfach **PROJEKT1**.



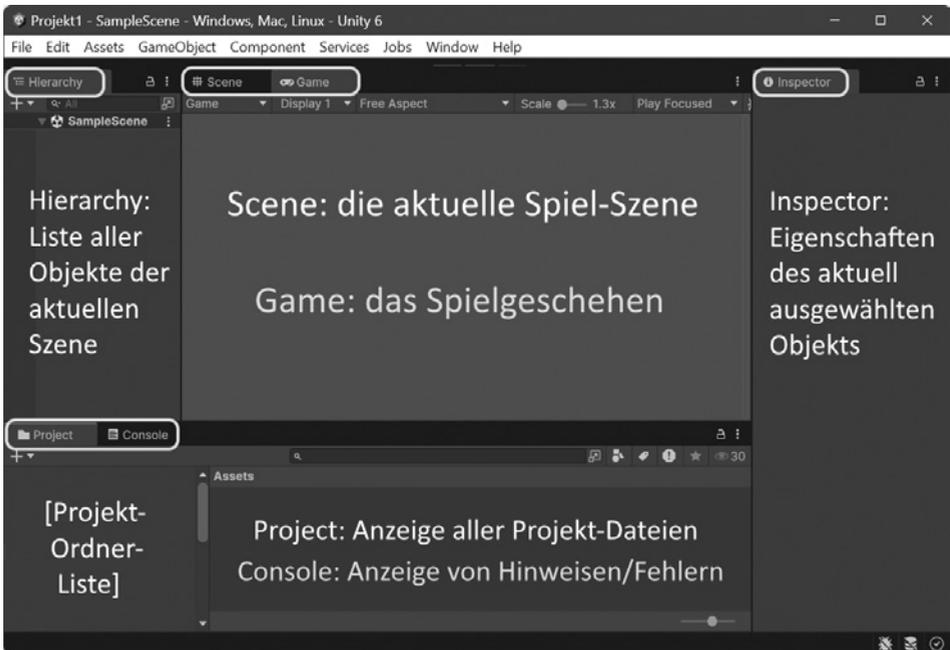
6. Klicken Sie zum Abschluss auf **CREATE PROJECT**.
Es dauert nun eine Weile, bis Ihr Projekt in der Liste unter **PROJECTS** auftaucht.

Kapitel 1

Das erste Projekt



Anschließend zeigt uns Unity endlich seine Arbeitsumgebung. Schauen wir uns erst einmal die Aufteilung der wichtigsten Fensterbereiche an:

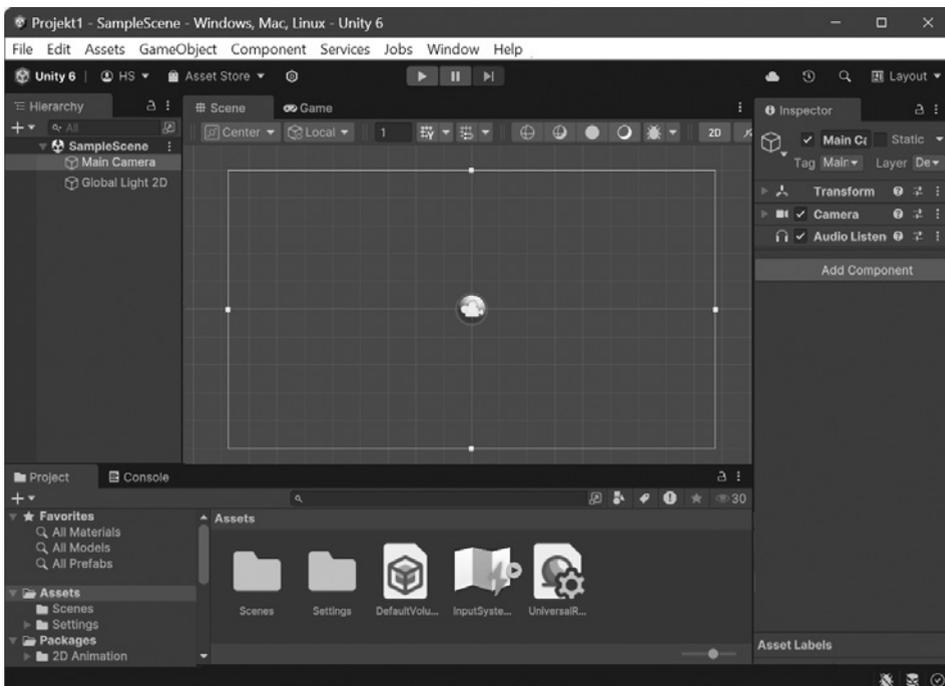


- Im GAME-Fenster ist es erst einmal leer. Da sehen Sie später Ihr Spiel in Echtzeit ablaufen, wenn Sie es durch einen der darüberliegenden Buttons gestartet haben.
- Dahinter liegt das SCENE-Fenster. Und es gibt auch schon zwei Objekte: Kamera und Licht. Doch für ein Spiel brauchen wir dann noch mindestens ein weiteres Objekt wie eine Kugel oder eine Figur.

- Im HIERARCHY-Fenster sind bis jetzt nur MAIN CAMERA und ggf. GLOBAL LIGHT aufgelistet. Dort stehen dann später auch alle Objekte, die zur Szene eines Spiels gehören (jedes Spiel könnte mehrere Szenen haben).
- Das PROJECT-Fenster erfasst die Ordner mit dem gesamten Zubehör für alle Spielszenen. Dazu gehören natürlich u.a. auch Programmteile. Bilder, die Sie als Spiel-Objekt einsetzen wollen (wie z.B. eine Kugel oder eine Figur), lassen sich einfach mit der Maus aus einem Ordnerfenster unter Windows hier hineinziehen. Damit wird die entsprechende Datei ins Projekt kopiert.
- Dahinter findet sich das CONSOLE-Fenster, das sich u.a. bei Fehlern meldet. Außerdem lassen sich dort Daten anzeigen, z.B. von Spiel-Objekten.
- Um sich die Eigenschaften eines Objekts nicht nur anzuschauen, sondern auch bearbeiten zu können, gibt es das INSPECTOR-Fenster. Damit werden wir des Öfteren zu tun haben.

Schalten Sie mal vom GAME-Fenster ins SCENE-Fenster um.

In der Mitte ist das Symbol für die Kamera.



Wenn Sie im HIERARCHY-Fenster auf MAIN CAMERA klicken, zeigt der INSPECTOR plötzlich eine ganze Menge an (ändern sollten Sie aber daran nichts).

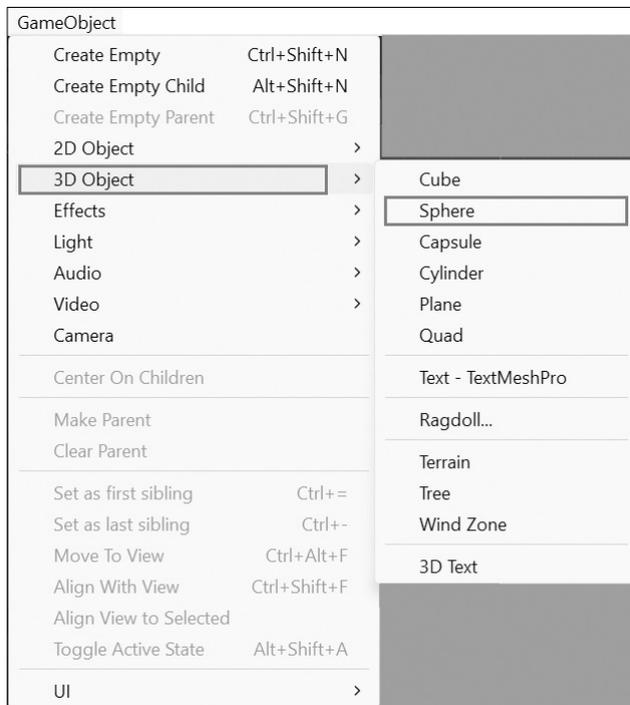
Mit dem Hauptmenü bekommen wir immer wieder zu tun. Die Bedeutung der meisten Menüpunkte klären wir nach und nach.



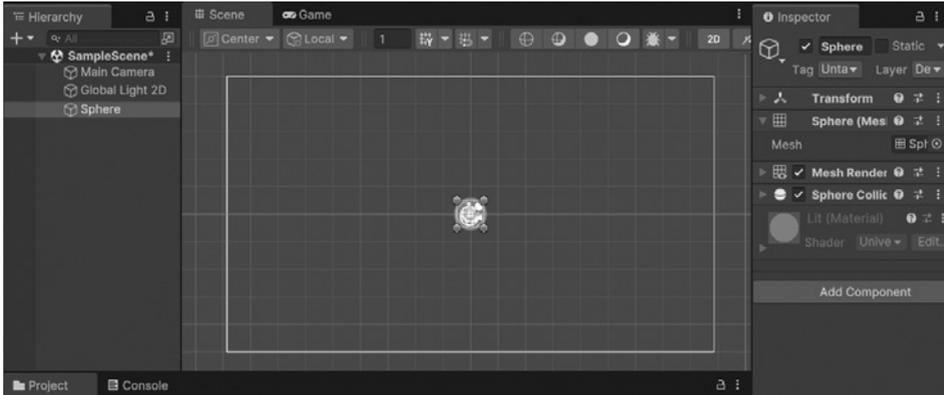
1.2 Ein Objekt zum Spielen

Wir beginnen mit etwas Einfachem. Dazu brauchen wir eine Kugel, und die soll sich über das Spielfeld bewegen lassen, z.B. mit der Maus oder mit den Tasten.

1. Klicken Sie oben im Hauptmenü auf **GAMEOBJECT** und dann auf den Eintrag **3D OBJECT**. Im Zusatzmenü bekommen Sie nun eine Auswahl. Klicken Sie auf den Eintrag **SPHERE** (= Kugel).



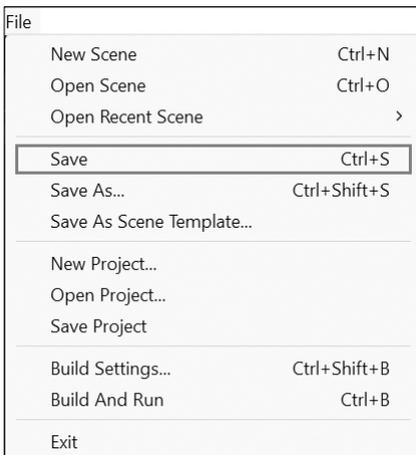
Anschließend taucht im **SCENE**-Fenster etwas auf, das bei genauerem Hinsehen wie ein Kreis aussieht – aber irgendwie auch recht mickrig. Außerdem zeigt der **INSPECTOR** zahlreiche Informationen über unser neues Spiel-Objekt.



Dargestellt sind in der Mitte 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.

2. Klicken Sie auf FILE und SAVE.

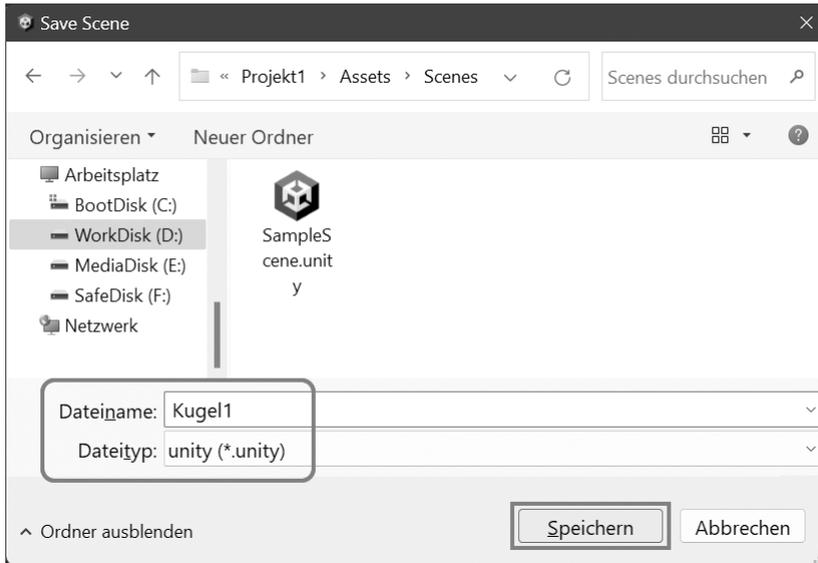


SampleScene

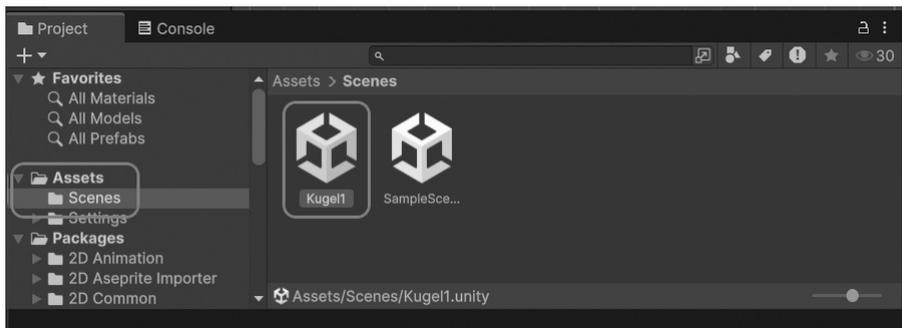
In Unity heißt diese Szene zuerst SAMPLESCENE. Passt Ihnen der Name nicht, müssen Sie die Option SAVE AS wählen und im Dialogfeld einen Namen eingeben, z.B. KUGEL1 (wenn Ihnen nichts Besseres einfällt). Die Kennung UNITY wird automatisch angefügt. Dann klicken Sie auf SPEICHERN.

Kapitel 1

Das erste Projekt



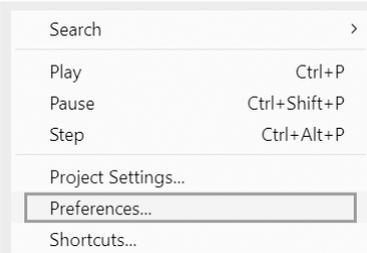
Wenn Sie anschließend im PROJECT-Fenster auf ASSETS klicken, sehen Sie den Ordner SCENES, darin befindet sich das Symbol für die Szenen-Dateien.



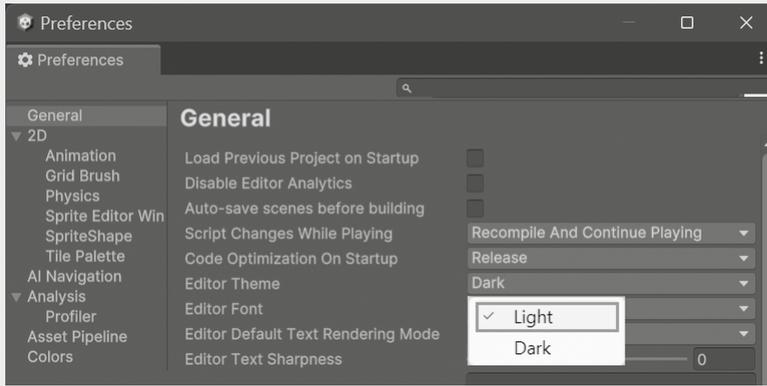
Dark oder Light

Bevor wir nun weitermachen, möchte ich Ihnen eine Möglichkeit vorstellen, die bisher dunkle Anzeige auf ein hellere umzustellen.

Suchen Sie im (sehr langen) EDIT-Menü nach dem Eintrag PREFERENCES und klicken Sie darauf.



Suchen Sie unter **GENERAL** den Eintrag **EDITOR THEME**. Dort wählen Sie **LIGHT** statt **DARK**.



Anschließend ist alles um einiges heller. Wenn es Ihnen gefällt, lassen Sie es so. Oder Sie kehren zurück zum Dark-Mode.

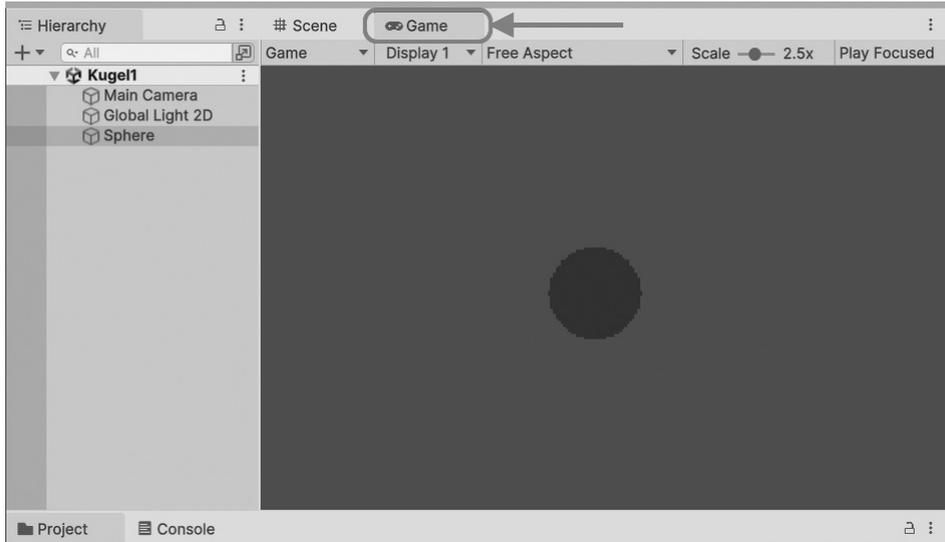
Ich werde ab jetzt hier im Buch den Light-Mode benutzen, weil die Abbildungen lesbarer sind. Sie selbst können frei wählen, ob Sie beim Dark-Mode bleiben oder auch in den Light-Mode wechseln.

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

1. Dazu schalten Sie mit Klick auf den Reiter mit dem Text **GAME** (direkt rechts neben dem **SCENE**-Reiter) die Anzeige um.

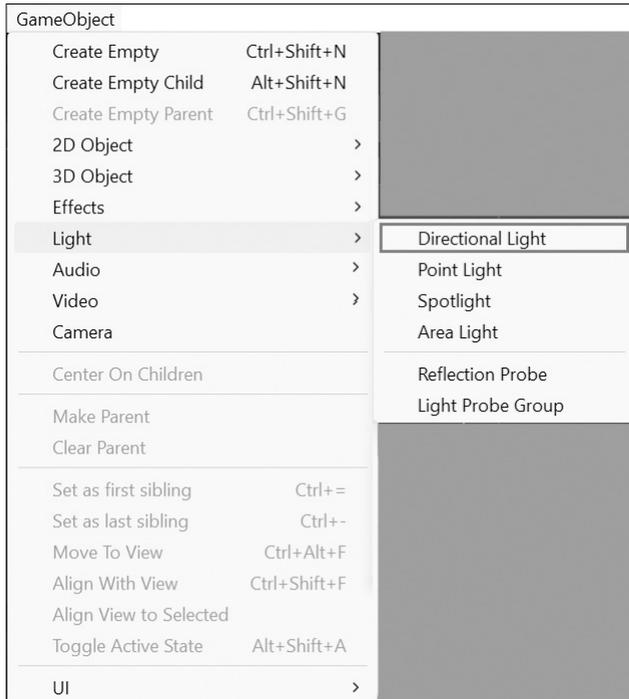
Kapitel 1

Das erste Projekt

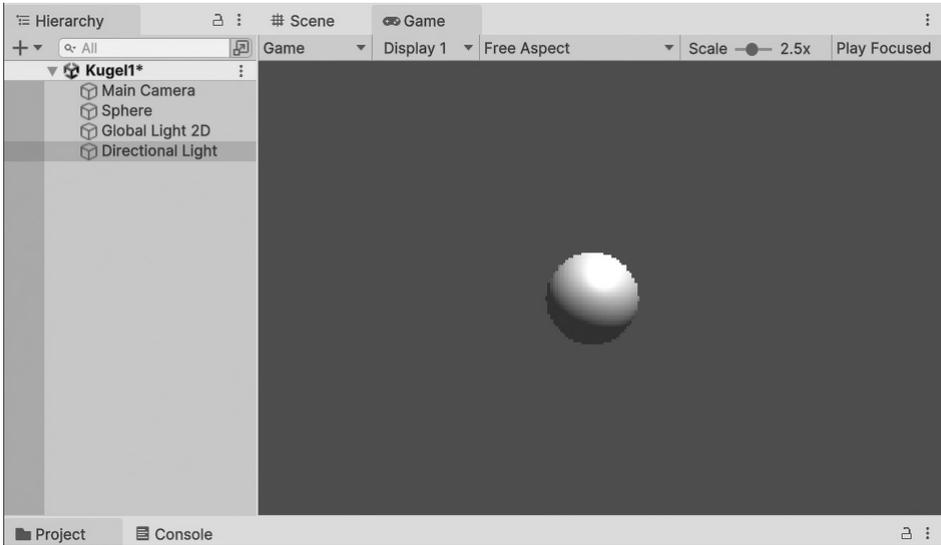


Im GAME-Fenster können Sie nun etwas sehen. Naja, wie eine Kugel schaut dieser schwarze Kreis (noch) nicht aus. Aber vielleicht lässt sich daran etwas ändern.

2. Klicken Sie im Hauptmenü auf GAMEOBJECT und dann auf LIGHT. Im Zusatzmenü wählen Sie den Eintrag DIRECTIONAL LIGHT.

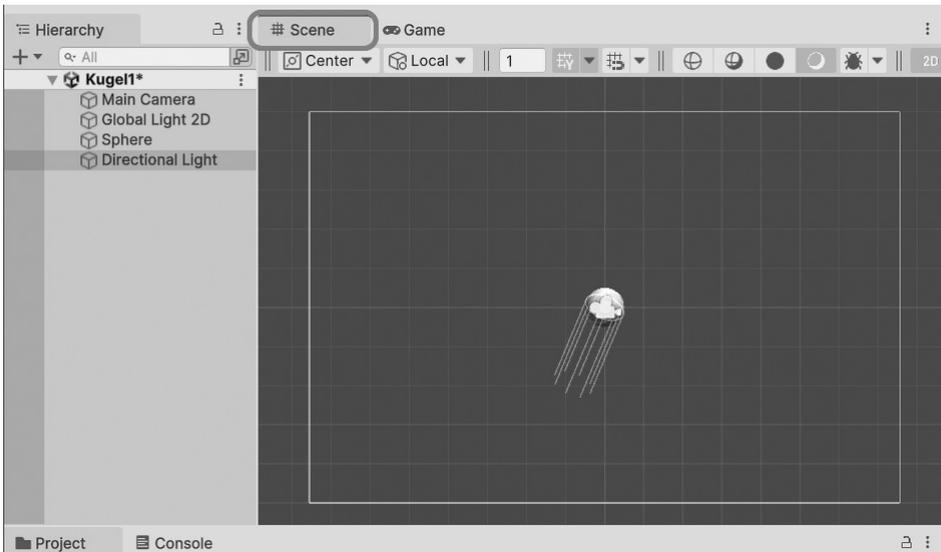


Kurz darauf sehen Sie die Kugel in einem anderen Licht. Erkennt man das nur im GAME-Fenster?



3. Schalten Sie doch mal um zum SCENE-Fenster.

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



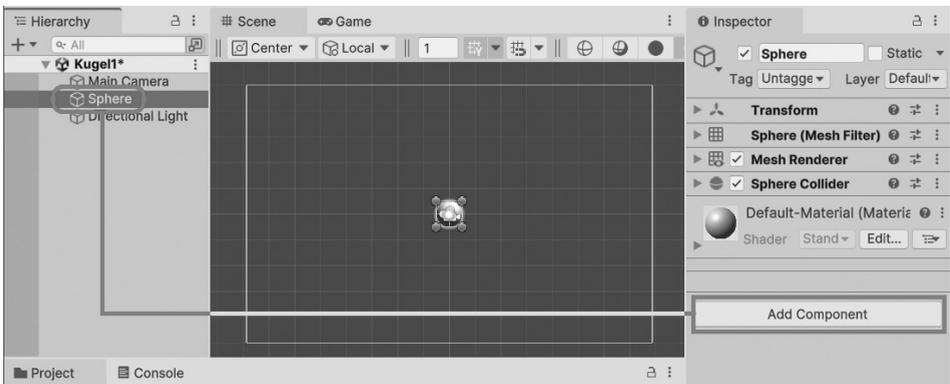
Nun gibt es ein weiteres Objekt, und es liegt ebenso wie die Kugel in der Fenstermitte.

Licht-Objekte

Wenn Sie wollen, können Sie GLOBAL LIGHT auch entfernen: Markieren Sie den Eintrag und drücken Sie die Taste `Entf`.

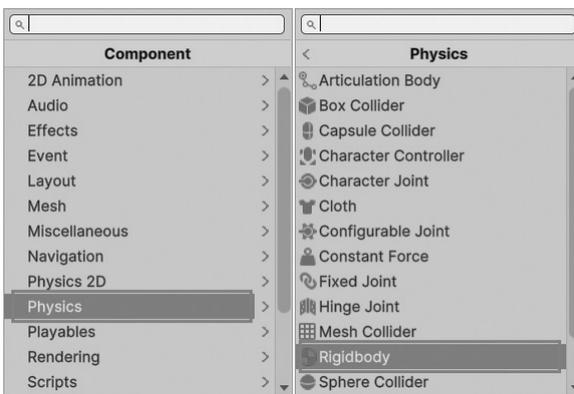
Toll wäre es, wenn sich die Kugel bewegen würde. Doch wie kriegen wir sie dazu? Zuerst einmal sollten wir diesem »Ding« physikalische Eigenschaften geben. Dass das Objekt aussieht wie eine Kugel, heißt noch nicht, dass es sich auch wie eine Kugel aus einem bestimmten Material verhält.

1. Markieren Sie jetzt im HIERARCHY-Fenster (links) den Eintrag SPHERE. Dann schauen Sie im INSPECTOR-Fenster (rechts) nach einem Button mit der Aufschrift ADD COMPONENT und klicken darauf.



Ein kleines Kontextmenü öffnet sich.

2. Wählen Sie den Eintrag PHYSICS. Und im nächsten Menü klicken Sie auf RIGIDBODY.



Hauptmenü

Ein alternativer Weg führt über das Hauptmenü: Dazu klicken Sie sich über COMPONENT und PHYSICS zu RIGIDBODY durch.

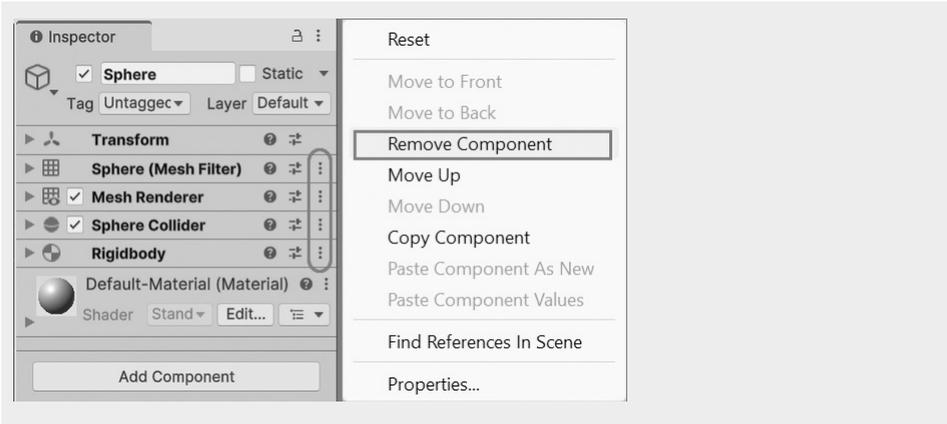


Damit bekommt die Kugel (Sphere) einige physikalische Eigenschaften wie *Masse* (unsere Kugel wiegt also auf einmal etwas) und *Gravitation* (sie wird vom Boden angezogen, würde also aus der Luft herunter auf den Boden fallen). Auch Kollisionen mit anderen Objekten und ihre Folgen sind nun möglich (ich gehe später genauer auf Einzelheiten ein).

Komponente wieder entfernen?

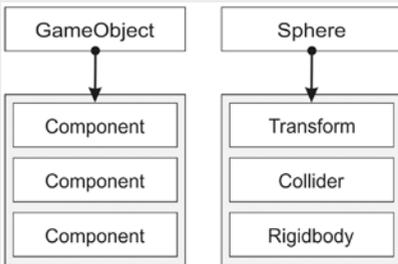
Falls Sie aus Versehen eine falsche Komponente hinzugefügt haben: Wie werden Sie diese wieder los? Schauen Sie im INSPECTOR-Fenster mal genauer hin. Hinter jedem Komponenten-Namen sind ganz rechts drei kleine Pünktchen.

Klickt man darauf, öffnet sich ein Kontextmenü, in dem u.a. der Eintrag REMOVE COMPONENT zu finden ist (außer bei der TRANSFORM-Komponente, die lässt sich nicht entfernen).



Spiel-Hierarchie

Wie Sie sicher bemerkt haben, gibt es in Unity diese Hierarchie: Eine *Szene* umfasst mindestens ein Objekt vom Typ `GAMEOBJECT`. Die Kamera ist ja schon beim Erzeugen eines Projekts vorhanden. Dazu kommt dann so etwas wie eine Spielfigur. In unserem Fall ist das erst mal nur eine Kugel. Die hat dann verschiedene Komponenten, ebenfalls Objekte, nur vom Typ `COMPONENT`.



Die Komponente `TRANSFORM` hat jedes Spiel-Objekt »von Geburt an«. Weitere Komponenten lassen sich (fast) beliebig hinzufügen (aber auch wieder entfernen).

Wir sollten das Ganze schon einmal ausprobieren.

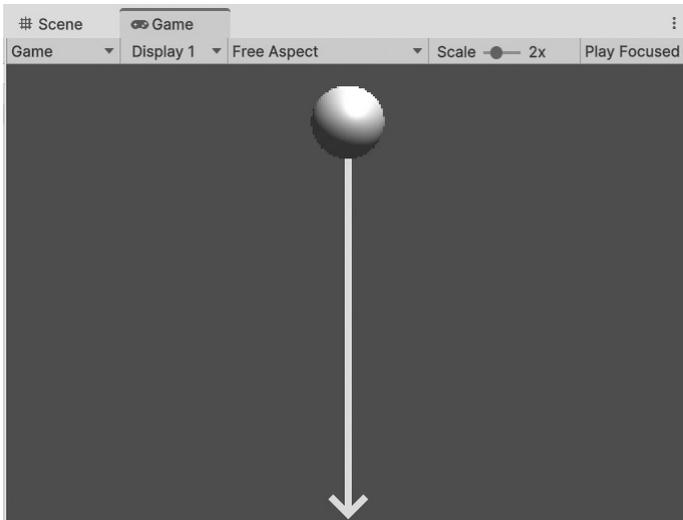
1. Wechseln Sie dazu ins `GAME`-Fenster.



2. Dann klicken Sie im Hauptmenü auf `EDIT` und suchen Sie den Eintrag `PLAY`. Oder Sie verwenden die Tastenkombination `[Strg] + [P]`.

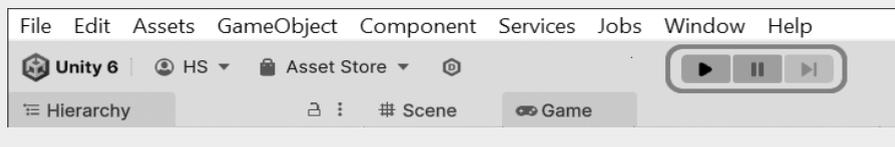


Im ersten Moment passiert anscheinend nichts, dann auf einmal fällt die Kugel und verschwindet aus dem Spielfeld.



Start und Stopp

Start und Stopp eines Unity-Spiels können Sie auch über Buttons steuern. Ganz oben, direkt unter der Menüzzeile, finden sich drei davon. Sie erinnern an die Steuerung z.B. bei Audio-Rekordern.



Das linke (mit dem Dreieck) ist der PLAY-Button. Per Mausklick lässt sich damit ein Spiel starten und stoppen. Mit dem mittleren Button können Sie das Spiel auch pausieren und dann weiterlaufen lassen.

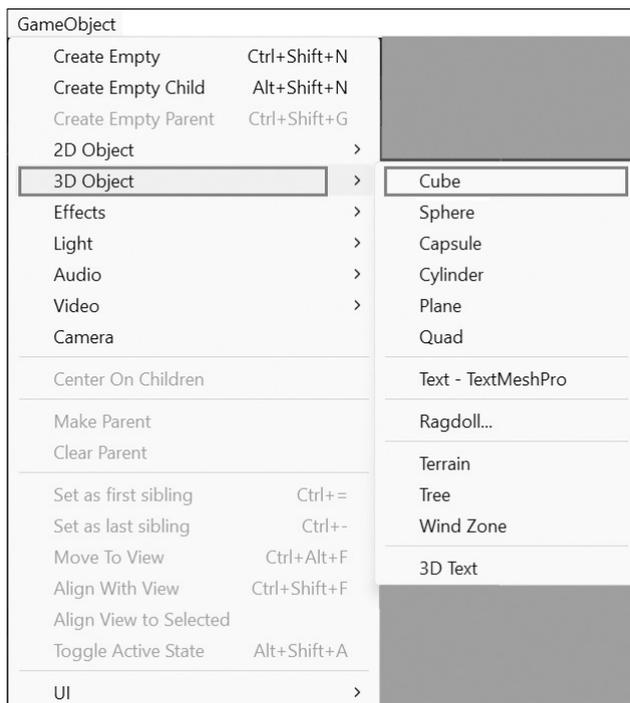
1.3 Gravitation und Kollision

Irgendwie muss es nicht sein, dass die Kugel gleich nach dem Start als Spielfigur aus dem sichtbaren Bereich herausfällt. Damit das nicht passiert, könnte man die Gravitation ausschalten.

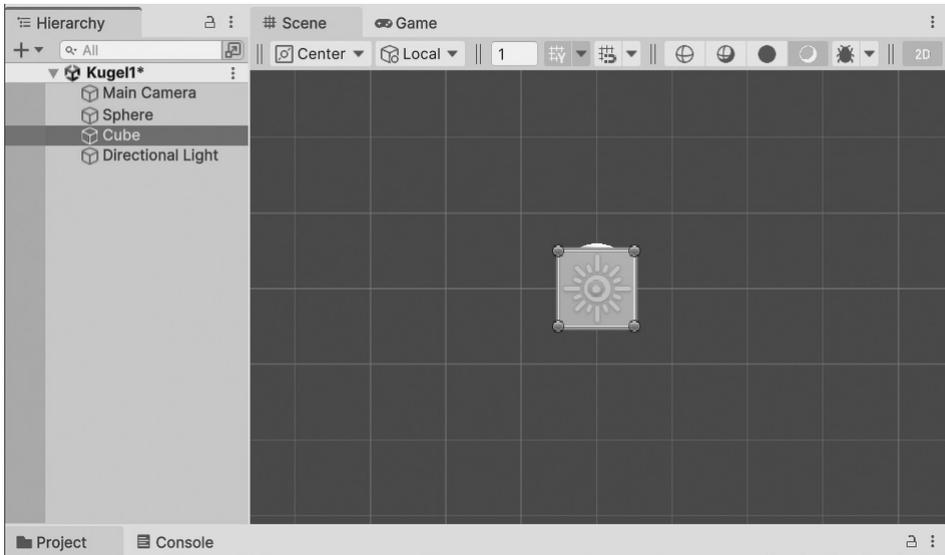
Andererseits kann die Gravitation einem beim Spielen nützlich sein. Denken Sie z.B. an ein Jump & Run-Game. Da geht es ja um Figuren, die springen und rennen, aber immer wieder irgendwo landen (und dazu brauchen sie die »Erdbziehungs-kraft«).

Eine andere und bessere Möglichkeit wäre es, für das Spielfeld eine untere Grenze zu verwenden. Das könnte ein Quader sein. Probieren wir's aus.

1. Klicken Sie im Hauptmenü auf `GAMEOBJECT` und dann auf `3D OBJECT`. Im Zusatzmenü suchen Sie diesmal den Eintrag `CUBE` (= Würfel) und klicken darauf.



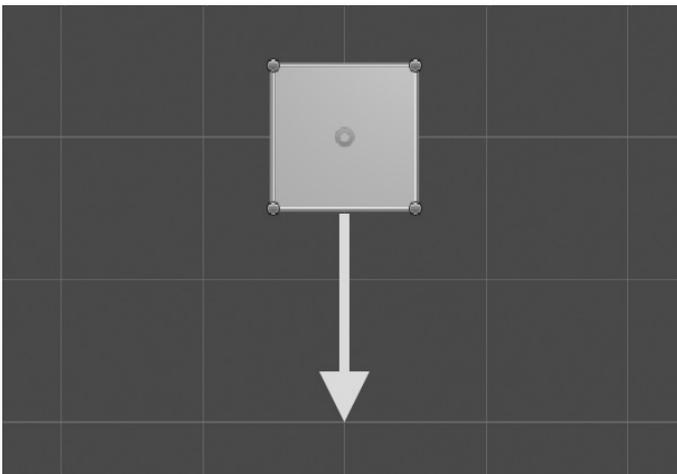
Im SCENE-Fenster hat sich nun über den Kreis so etwas wie ein Quadrat gelegt. Auch hier zeigt der INSPECTOR zahlreiche Informationen über das neue Spiel-Objekt.



2. Vielleicht ist es sinnvoll, die Szene erst einmal wieder speichern (mit Klick auf FILE UND SAVE).

Aktuell ist der Quader noch ein Würfel, später könnte man daraus eine Platte machen. Doch erst einmal suchen wir eine Möglichkeit, den Quader nach unten zu verschieben. Grundsätzlich gibt es da zwei Möglichkeiten:

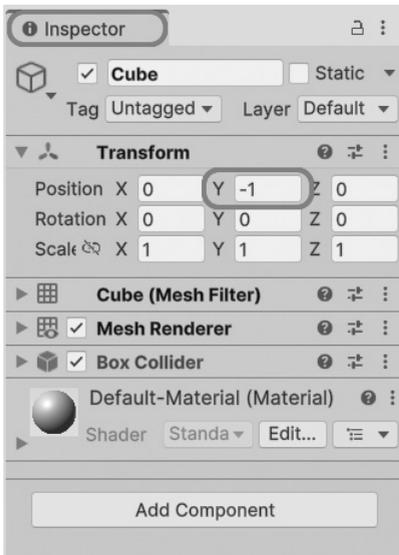
- Man packt das Objekt im SCENE-Fenster mit der Maus und zieht es nach unten (das Ganze geht natürlich auch in andere Richtungen).



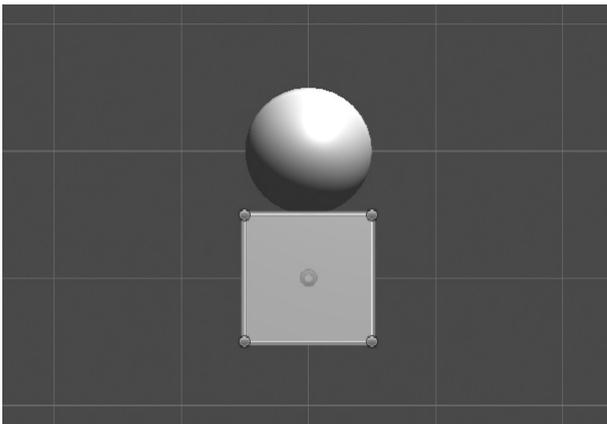
Kapitel 1

Das erste Projekt

- Man ändert die Werte für POSITION im INSPECTOR-Fenster. Genauer: den y-Wert auf -1.



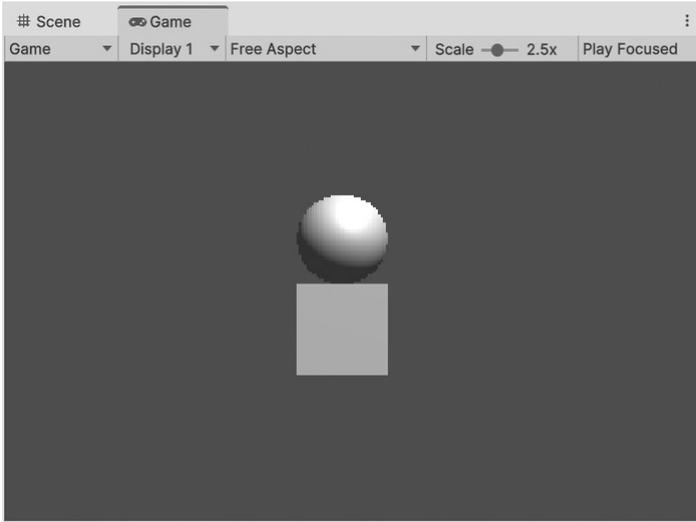
3. Sorgen Sie dafür, dass der Quader direkt unter der Kugel liegt.



4. Und nun können Sie sich im GAME-Fenster anschauen, was passiert, wenn Sie das Spiel starten (damit man mehr sehen kann, habe ich SCALE auf 2,5x eingestellt).

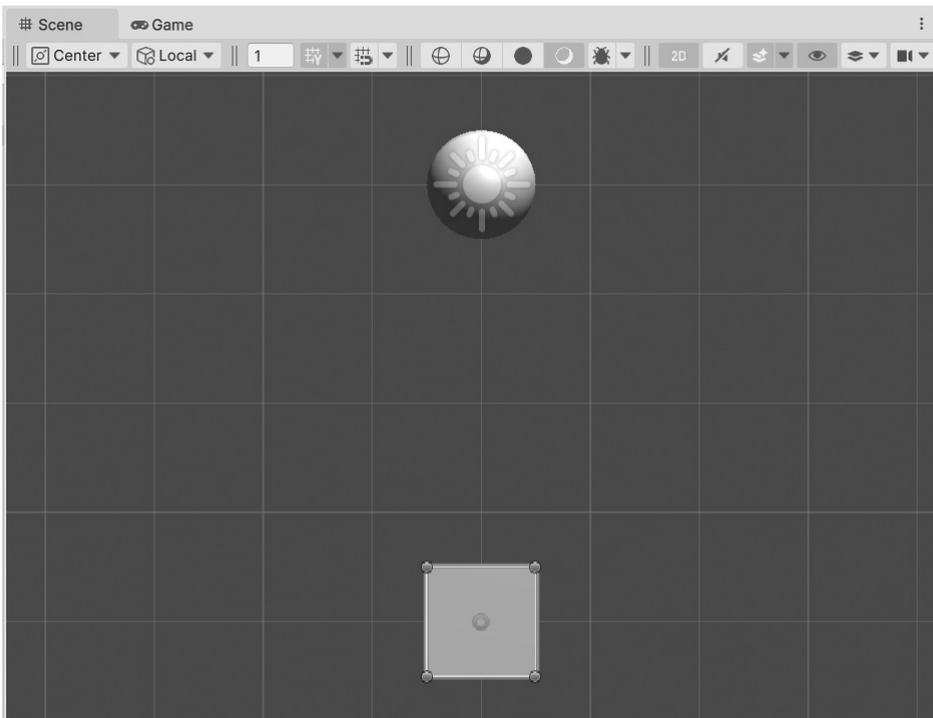
Zoom

Man kann auch mit dem Mauszeiger ins SCENE-Fenster fahren und das Rollrad der Maus zum Zoomen benutzen.



Nichts passiert? Ja und nein. Die Kugel versucht wohl zu fallen, wird aber vom Quader aufgehalten. Der verhindert, dass hier die Gravitation sichtbar wird. Denn die Kugel liegt auf dem Quader.

5. Verschieben Sie nun den Quader bis zum Spielfeldrand nach unten. Achten Sie darauf, dass er möglichst genau unter der Kugel liegt.



Kapitel 1

Das erste Projekt

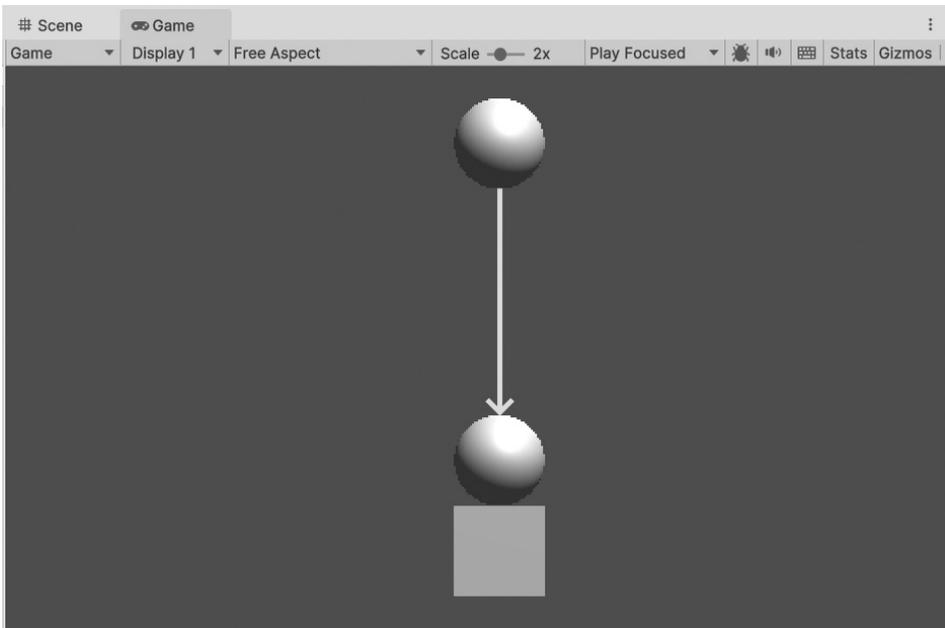
Bei mir steht im INSPECTOR-Fenster hinter POSITION der y-Wert -5. Bei Ihnen kann da natürlich auch etwas anderes stehen.

Verschiedene Positionen?

Vielleicht haben Sie hin und wieder den Eindruck, dass die Position eines Objekts im SCENE-Fenster anders ist als im GAME-Fenster. Das hängt mit der Kamera zusammen. Die allein bestimmt, was im GAME-Fenster wo zu sehen ist.

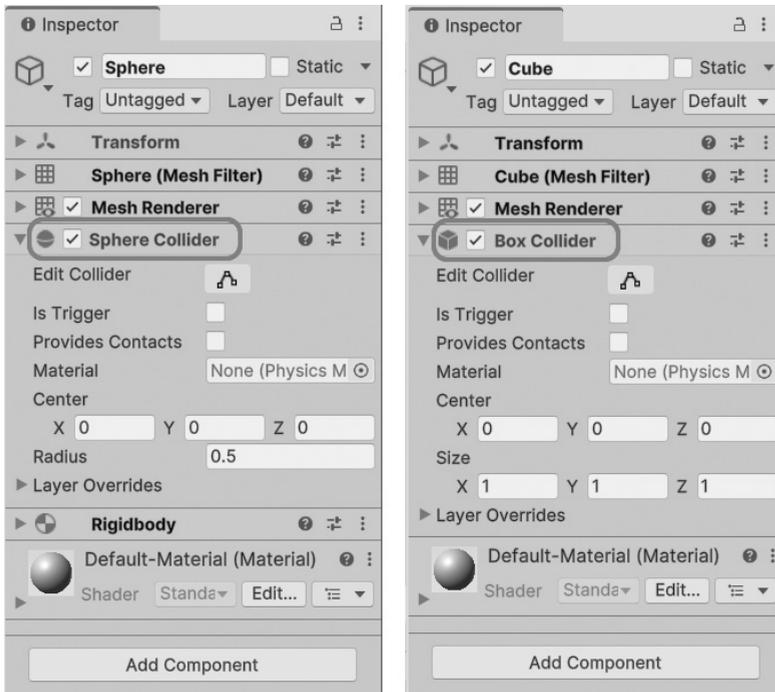
Wenn Sie wollen, können Sie die Kamera (MAIN CAMERA) mal anklicken, damit sie markiert ist. Dann lässt sie sich verschieben, ebenso wie eine Kugel oder ein Quader. Und damit ändert sich auch die Perspektive im GAME-Fenster. Über EDIT/UNDO oder `[Strg]+[Z]` lässt sich diese Verschiebung wieder rückgängig machen.

6. Starten Sie das Spiel nun erneut und schauen Sie zu, wie die Kugel fällt und auf dem Quader landet – eigentlich wie zu erwarten, oder?



Dass dies keine Selbstverständlichkeit ist, werden Sie gleich sehen. Verantwortlich dafür, dass die Kugel auf dem Quader landet und nicht weiterfällt, ist nicht die Gravitation, sondern eine andere Eigenschaft, die sie von Anfang an hatte – ebenso wie der Quader.

Schauen wir mal ins INSPECTOR-Fenster. Dort finden wir unter den Eigenschaften (auch Komponenten genannt) jeweils einen Sphere-Collider und einen Box-Collider für Kugel beziehungsweise Quader.



Kollisionslos

Machen wir mal einen Test: Wenn Sie das Häkchen vor einem dieser beiden Einträge entfernen und dann das Spiel erneut starten – z.B. mit dem Play-Button (oder über EDIT und PLAY), dann können Sie beobachten, dass die Kugel nun einfach durch den Quader hindurchfällt, ein Objekt ist für das andere sozusagen »Luft«.

Collider

Was ist ein *Collider*? Das hat etwas mit Kollision zu tun. Wenn zwei Objekte aufeinandertreffen, dann spricht man von einer Kollision. Das kann eine sanfte oder eine harte Kollision sein. Das Verhalten bei einer solchen »Begegnung« wird in Unity über Collider gesteuert.

Alle betroffenen Objekte müssen also Collider haben. Dabei hat nicht jedes Objekt den gleichen Collider-Typ. Jeder Collider lässt sich aktivieren und deaktivieren. Das

ist nützlich, denn manchmal soll keine Kollision stattfinden, dann lassen sich die Collider ausschalten.

1.4 2D oder 3D?

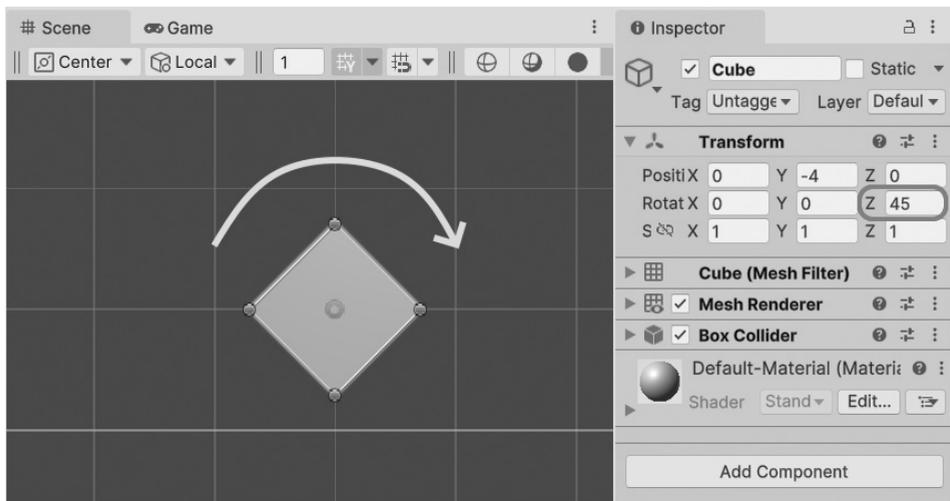
Halten wir jetzt erst einmal inne und schauen uns im INSPECTOR-Fenster mal genauer um. Und zwar auf das, was unter TRANSFORM steht. Ich habe das in einer Tabelle zusammengefasst. So sieht es für ein Objekt aus, das gerade erzeugt wurde:

	X	Y	Z	Mögliche Aktion
POSITION	0	0	0	Verschieben
ROTATION	0	0	0	Drehen
SCALE	1	1	1	Größe ändern

Es gibt hier drei Möglichkeiten, etwas mit einem Objekt »anzustellen«. Die erste Möglichkeit haben Sie bereits kennengelernt, als Sie den Quader nach unten verschoben haben.

Die zweite Möglichkeit, ein Objekt zu drehen, brauchen wir jetzt noch nicht. Bei einer Kugel sieht man davon nichts (es sei denn, das Licht ändert sich mit), bei einem Quader schon.

1. Da sollten Sie gleich mal ausprobieren: Setzen Sie im INSPECTOR-Fenster unter ROTATION für Z z.B. den Wert 45 ein.

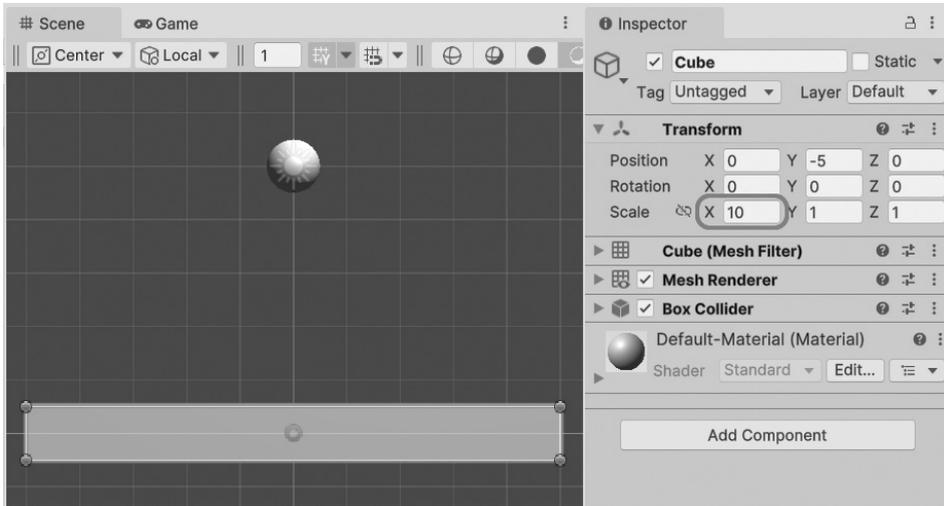


Und schon wird der Würfel um 45 Grad gedreht.

Die dritte Möglichkeit der Transformation eines Objekts ist die Veränderung der Maße (Skalierung). Davon wollen wir jetzt Gebrauch machen. Denn der Quader

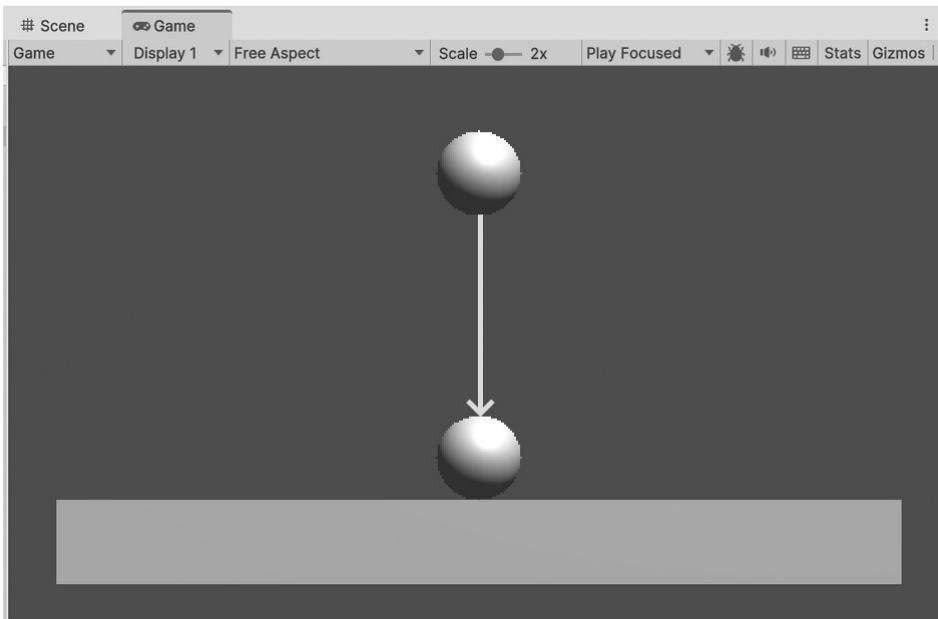
sieht ein bisschen mickrig aus. Warum machen wir aus ihm nicht einen Balken, der den ganzen unteren Spielfeldrand abdeckt?

2. Machen Sie zuerst die Drehung wieder rückgängig, dann ändern Sie im INSPECTOR-Fenster unter SCALE für X den Wert – mit einer Zahl zwischen 10 und 15.



Damit ist der Quader sozusagen die Bodenplatte, auf die die Kugel fällt.

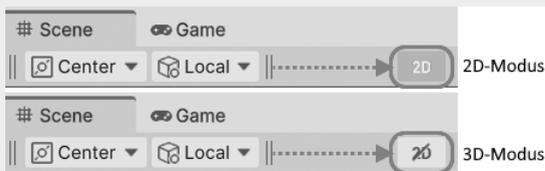
3. Speichern Sie nun die Szene, dann starten Sie das Spiel und schauen zu.



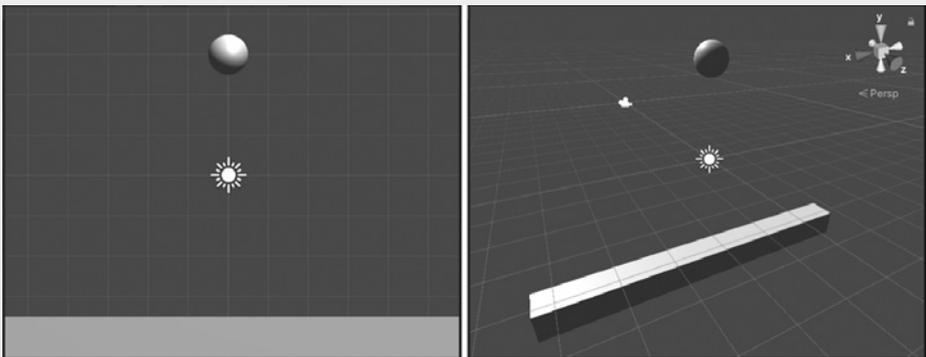
Ist das jetzt 2D oder 3D? Diese Frage haben Sie sich vielleicht schon viel früher gestellt. Genau genommen haben wir es die ganze Zeit mit Objekten zu tun, die dreidimensional sind. Da wir uns erst mal nur im 2D-Bereich (hier auf einer Fläche mit Höhe und Breite) bewegen wollen, haben wir beim Erzeugen des Projekts die *2D-Ansicht* eingeschaltet.

2D oder 3D

Man sieht das im SCENE-Fenster an dem kleinen »eingedrückten« 2D-Button (der liegt weiter rechts):



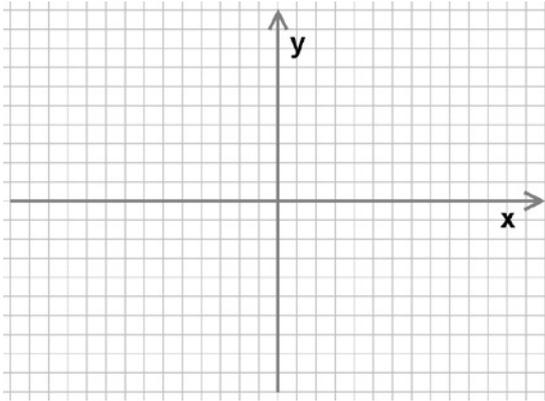
Hier lässt sich die Anzeige zwischen 2D und 3D umschalten. Im 3D-Modus ist die Anzeige 2D durchgestrichen.



Bei der Kugel sieht man den Unterschied nicht so stark, beim Quader schon, der wird in 2D nur als blasses Rechteck dargestellt. (Würde man das Licht »ausschalten«, dann wäre die Kugel in 2D auch nur ein Kreis.)

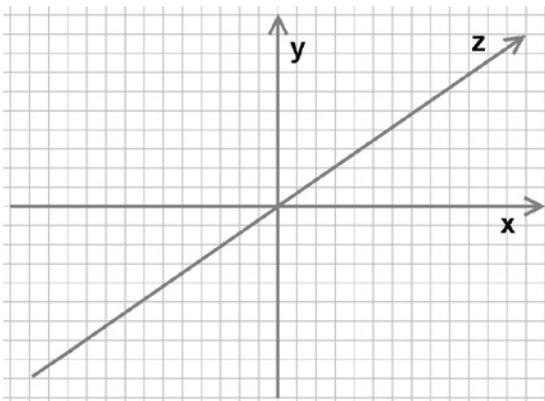
Das klassische Koordinatensystem besteht normalerweise aus der x-Achse (Horizontale bzw. Waagerechte) und der y-Achse (Vertikale bzw. Senkrechte).

Entlang der x-Achse geht es also nach links oder rechts, entlang der y-Achse nach oben oder unten. Der *Ursprung* befindet sich genau in der Mitte. Der Punkt dort hat die Koordinaten (0 | 0).



Das ist auch in Unity so, doch weil Unity auch ein System für 3D-Spiele ist, reichen keine zwei Achsen. Sondern es muss da noch eine dritte geben, z-Achse genannt. An der entlang geht es nach vorn oder nach hinten.

Schaut man von vorn auf das Koordinatensystem, dann kann man diese Achse nicht sehen. Um alle Achsen dennoch in 2D sichtbar zu machen, greift man zu einem optischen Trick: Die z-Achse wird dann als Diagonale dargestellt.



Wenden wir uns wieder dem INSPECTOR-Fenster zu, dort sind ja unter TRANSFORM alle drei Koordinaten aufgeführt. Wenn man dort unter POSITION und SCALE z.B. für den Quader den Wert hinter Z ändert, wird man bei 2D-Ansicht im SCENE-Fenster nichts davon bemerken (außer wenn bei SCALE der Wert von Z = 0 wäre). In Wirklichkeit aber verschiebt sich der Quader nach hinten oder nach vorn oder er dehnt sich in diese Richtungen aus.

Wenn Sie Lust zum Experimentieren haben, dann schauen Sie der Kugel noch einmal beim Fallen zu, nachdem die »Bodenplatte« (bei gleicher Größe) nach vorn oder hinten verschoben wurde. Ergebnis: Die Kugel fällt weiter, weil sie ja nicht mehr auf den Quader trifft.

Kapitel 1

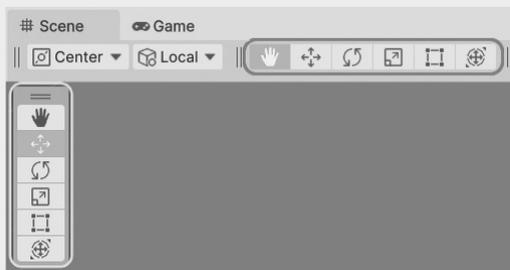
Das erste Projekt

	X	Y	Z	Ergebnis
POSITION	0	0	1	Hinter der Kugel
ROTATION	0	0	0	
SCALE	10	1	10	So lang wie breit

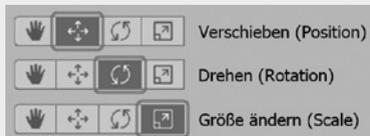
Sobald aber der Quader nach hinten und vorn ausreichend zu einer großen Plattform vergrößert wird, bekommt die Kugel wieder »Boden unter den Füßen«.

Transformationen mit der Maus

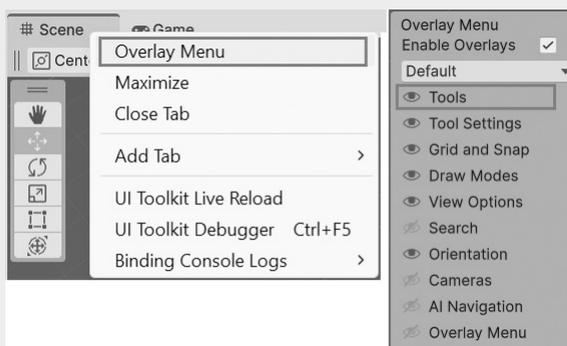
Wie man ein Objekt mit der Maus verschiebt, wissen Sie ja im Prinzip. Allerdings ist dazu eine bestimmte Einstellung nötig. Sie finden links oben im SCENE-Fenster oder direkt unter dem Hauptmenü eine Reihe von Buttons.



Darüber kann man Objekte mit der Maus verschieben (POSITION-Modus), die Größe ändern (SCALE-Modus) oder das Objekt drehen (ROTATION-Modus).



Die Umschaltung geht auch mit den Tasten **[W]** = Verschieben, **[E]** = Drehen, **[R]** = Skalieren.



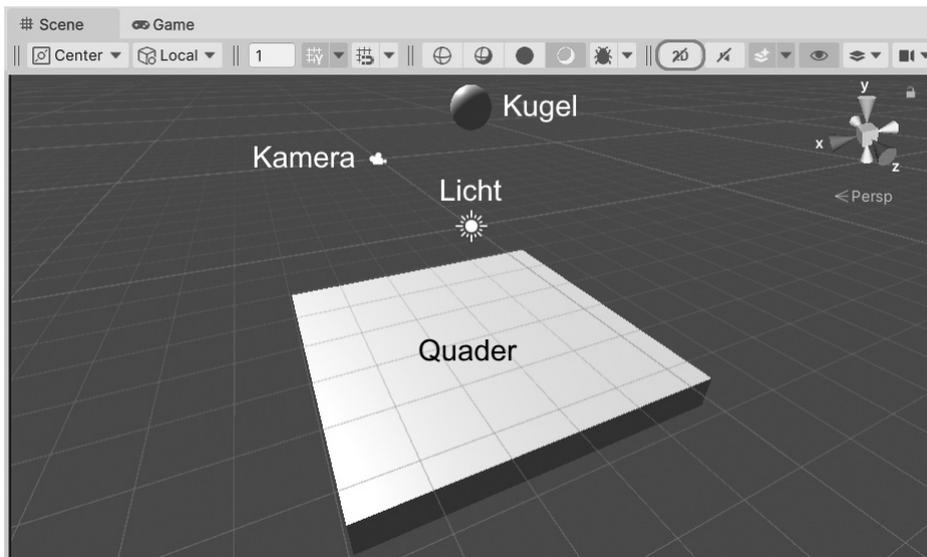
Falls Sie diese Buttons mal nicht finden oder sehen, lassen sie sich so wiederherstellen: Klick mit der rechten Maustaste auf den SCENE-Reiter, im Menü den Eintrag OVERLAY-MENU und dann im nächsten Menü auf TOOLS.

Auch wenn wir fürs Erste im 2D-Bereich bleiben werden, kann es nicht schaden, mal einen genaueren Blick auf die 3D-Ansicht zu werfen.

1. Klicken Sie oben (rechts) auf die Schaltfläche 2D.



Die 3D-Ansicht der Szene hatten wir ja weiter oben schon mal. Man sieht die Kugel nicht in der Mitte liegen. Die Kamera schaut aus einiger Entfernung von weiter hinten zu. Beim Quader, den ich »Bodenplatte« genannt habe, sieht man die Ausdehnung in die Tiefe. Und ganz oben rechts wird die aktuelle Perspektive angezeigt.

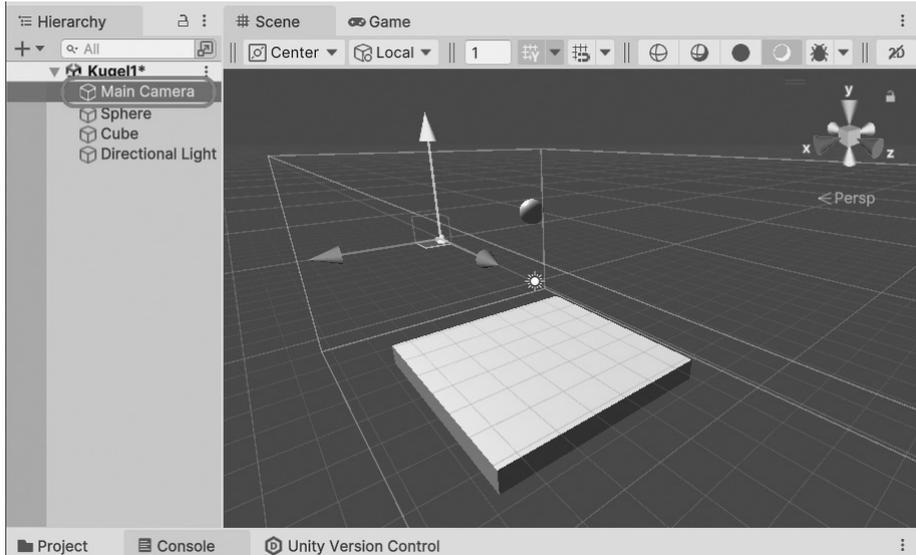


Um das Ganze auch mal im GAME-Fenster zu sehen, müssen wir die Kamerasicht im INSPECTOR-Fenster umschalten.

2. Markieren Sie dazu den Eintrag MAIN CAMERA links im HIERARCHY-Fenster.

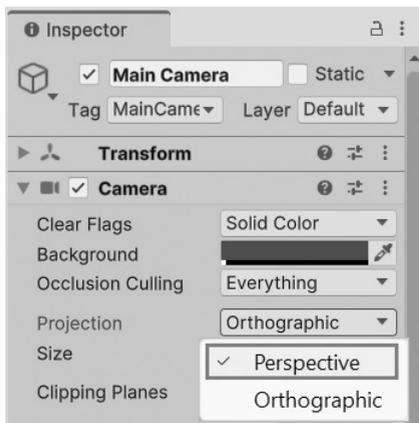
Kapitel 1

Das erste Projekt

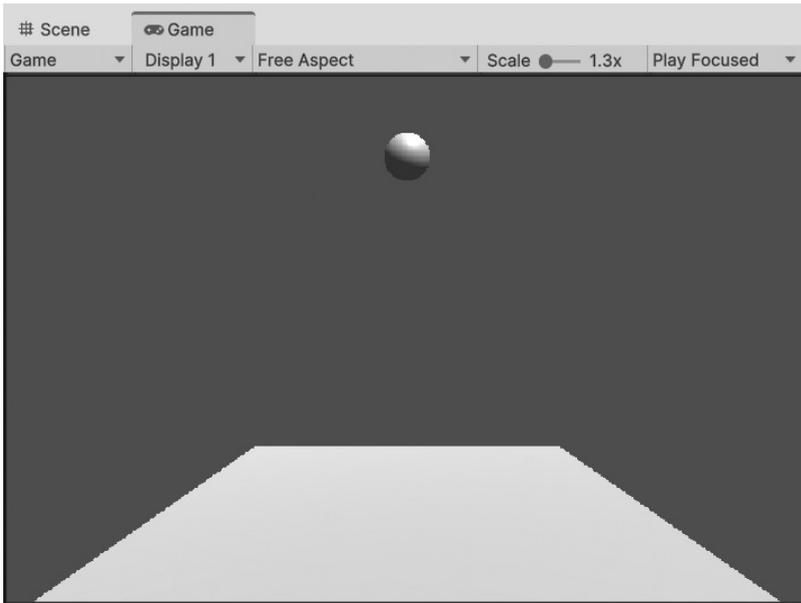


Würde ich jetzt ins Game-Fenster umschalten, wäre dort aber noch immer die 2D-Ansicht zu sehen.

3. Deshalb klicken Sie jetzt rechts im INSPECTOR-Fenster hinter PROJECTION auf ORTHOGRAPHIC. In dem kleinen Zusatzmenü wählen Sie den Eintrag PERSPECTIVE.



4. Wechseln Sie nun zum GAME-Fenster, dann sieht man den kompletten Boden.



Allzu toll sieht es nicht aus, auch weil alles nur weiß bis grau auf blauem Hintergrund zu sehen ist.

5. Nicht nur deshalb sollten Sie wieder auf die ORTHOGRAPHIC-Ansicht zurückgehen. Denn wir bleiben ja erst einmal im 2D-Bereich.

1.5 Ausblick

Unser erstes kleines Projekt ist damit fertig. Nichts Besonderes, aber auch nicht übel für den Anfang. Zuletzt sollen Sie noch wissen, wie man den Spiel-Objekten einen anderen Namen geben kann. Dazu muss das jeweilige Objekt im HIERARCHY-Fenster markiert sein.

6. Markieren Sie das jeweilige Objekt im HIERARCHY-Fenster, zum Beispiel SPHERE. Dann drücken Sie die Taste **[F2]**.

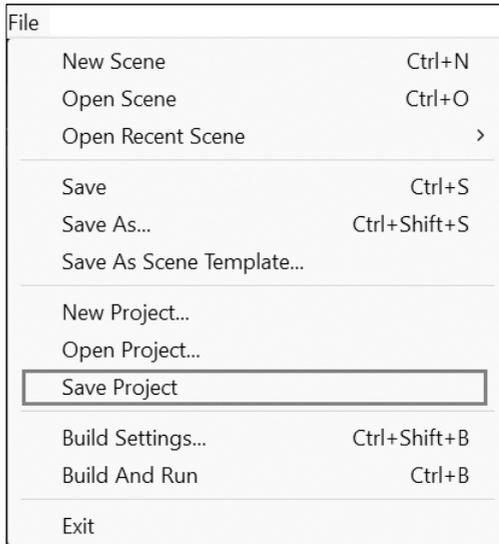


Nun kann man durch direktes Neueintippen den alten Namen überschreiben. Bei mir heißt die Kugel nun »Kugel« und der Quader bekommt den Namen »Boden«.

Kapitel 1

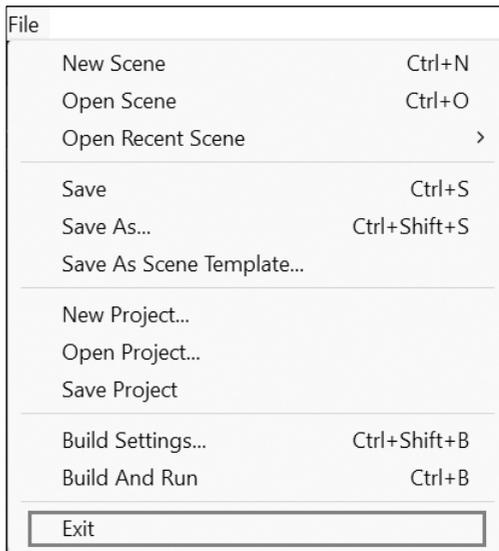
Das erste Projekt

Speichern Sie dann am besten das ganze Projekt:



Und damit wird es Zeit, Unity erst mal wieder zu verlassen.

7. Dazu klicken Sie auf FILE und dann auf EXIT. Oder Sie klicken im Hauptfenster ganz oben rechts auf das kleine X – wie auch bei anderen Programmen üblich.



Damit wäre eine Verschnaufpause fällig. Sie wissen nun schon, wie man ein (kleines) Projekt erstellt und dass dies aus mindestens einer Szene und einem Spiel-

Objekt besteht. Und Sie wissen auch, wie man ein Objekt erzeugt (über `GAMEOBJECT`) und eine Komponente hinzufügt (über `COMPONENT`).

Hier nochmal einige wichtige Elemente eines Spiel-Projekts im Überblick:

<code>GAMEOBJECT</code>	Spiel-Objekt, wie z.B. Quader oder Kugel, aber auch Kamera oder Licht
<code>COLLIDER</code>	Komponente für Kollisionen
<code>RIGIDBODY</code>	Komponente für physikalische Eigenschaften, wie z.B. Masse oder Gravitation

Für jedes Objekt gibt es eine *Transformations*-Komponente mit diesen Optionen:

<code>TRANSFORM</code>	Komponente für Änderungen der Lage und Größe
<code>POSITION</code>	Verschieben in alle Richtungen (3D)
<code>ROTATION</code>	Drehen in alle Richtungen (3D)
<code>SCALE</code>	Vergrößern/Verkleinern in alle Richtungen (3D)

Stichwortverzeichnis

Symbole

// 56

1st Person 121

2D-3D 13, 34

2D-Ansicht 37, 367

3D-Ansicht 37

3rd Person 121

A

Absolutwert 330

AddForce 52, 199

AI 291

Alpha Clipping 244

Alpha-Wert 252

Anchor 370

Android 392

Angle 365

Animation

Attacking 324

Clip 271, 309

Datei 309

Default 313

Dying 310

erzeugen 273

Fenster 309

Keyframe 274

Moving 275

Ordner 269

Position 272

Rotation 272

Speed 319

Trigger 280

Zwischenwerte 275

Animator 271

Condition 316

Controller 271, 312

enabled 281

Fenster 312

Parameter 314

speed 281

State 312

Transition 315

AnimatorTrigger 280

Area Light 226

Asset 43

exportieren 64

importieren 66

Package 140

Asset Store 145, 225

attachedRigidbody 200

Attack 336

AttackControl 337

Audio

Clip 356

Formate 354

Hintergrund 360

Play 357

PlaySound 356

Stop 356

Audio Listener 354

AudioSource 355

Avatar 271

Axes 93

B

back 55

Bauelement 206

Baum

Einstellungen 171

entfernen 172

ersetzen 172

hinzufügen 170

Kollision 173

Baumaterial 203

Baum-Werkzeug 168

Behaviour 46

Bend Factor 190

Benutzerschnittstelle 367

Blender 160, 225, 265

Boden

Ausrichtung 292

Box Collider 186

Button

Negative 94

Positive 94

C

C# 46

CameraSwitch 138

Canvas 368

Capsule 131

Capsule Collider 186

CharacterController 80, 126

Chaser 336

Child 134, 369

class 50

ClimbControl 237

Climber 238

Cluster 175

- Collider 31, 186
 - anpassen 250
- Collider Trigger 113
- CollisionControl 199, 306
- Color 114
- Component 22
- Conditions 316
- Cone 348
- Console 409
- Container 211, 224
- Cos 364
- Create New Clip 310
- CreatureAudio 359
- CreatureControl 290
- CreatureHealth 372, 379

- D**
- Dark-Mode 18
- Debug 130
 - Log 408
- Deklaration 83
- deltaTime 60, 357
- Destroy 282, 307
- Detail-Werkzeug 174
- Dezimal-f 82
- Directional Light 226
- down 55
- Drehung
 - Ausgangspunkt 364
- Drop Height 389
- Dummy 130, 265
- Du-Perspektive 121

- E**
- Editor 47, 48
- Ellipsoid 266
- else 103, 242
- Eltern-Objekt 369
- Emission 347
- Empty Object 186, 210, 249
- Empty Trigger 235
- enabled 139

- Energie
 - Anzeige 375
 - Kontrolle 373
 - Verlust 377
 - wiederherstellen 390
- Ereignis-Methode 114, 200
- Ersetzen 380
- Er/Sie-Perspektive 121
- Escape 395
- eulerAngles 364
- EventSystem 368
- Export 64, 216

- F**
- false 139
- Farbänderungen 326
- f-Dezimal 82
- Fehler 79, 409
- Find (Objektname) 238, 258
- First Person 121
- FixedUpdate 60
- Flatten 247
- float 82
 - f 127
- Fog 251
- fogColor 252
- fogDensity 252
- forward 55
- Frame 60
- Freeze 112
- Fullscreen 393

- G**
- Game
 - Fenster 20
 - Play 25
 - Quit 395
- Game Build 391
- Game-Engine 9
- GameObject 16, 72
- Game Over 382
- Gamepad 95

- Game-View maximieren 168
- Gegner
 - erstellen 265
- Geräusche 353
- Gestaltungs-Werkzeug 151
- Gesundheit 366
- GetAxis 92
 - Maus 128, 181
 - Tasten 94
- GetButton 97
- GetComponent 52, 83
- GetKey 53
- GetKeyUp 246
- GetTouch 95
- Gizmo 137
- Gras
 - Einstellungen 175
- Gravitation 26
- Grenze
 - unsichtbar 187
- GUI
 - Canvas 368
 - Element 366
 - Image 367
 - Text 372

- H**
- Halbdrehung 246
- Hand-Symbol 193
- Health 372
- Height Mesh 292
- Hierarchie 24
- Hintergrund-Sound 360

- I**
- Ich-Perspektive 121
- if 53
- Import 66, 216
- Input 52, 126
- Input Axes 93
- Input Manager 92, 93
- int 82, 318

- Invoke 328, 338
 isClimbing 240
 isGrounded 97
 isKinematic 201
 isPlaying 353
- J**
- Joystick 95
 Jump 97
 Jump Distance 389
 Jump & Run 71
- K**
- Kamera
 - bewegen 122
 - drehen 126
 - springen 126
 - umschalten 135
 - verbinden 133, 136
 - Verknüpfung 139
 - verschieben 125
 KeyDown 246
 Keyframe 274
 KeyUp 246
 Kind-Objekt 369
 Kinematik 298, 320
 Klammer
 - geschweift 50, 51
 - rund 50
 Klettern 238
 Kletter-Trigger 236
 Kollision 26, 199
 - Aufprall 307
 - Bäume 173
 - Schaden 306
 Kommentar 56
 Komponente
 - entfernen 23
 - Rigidbody 23
 Kontextmenü 23, 132
 Kontrollstruktur 53, 103
 Konvertierung 405
- Koordinatensystem 34
 Korrektur 292
 Kreatur
 - Beine 267
 - Container 268
 - erstellen 265
 - Körper 267
 - Rigidbody 269
 Künstliche Intelligenz 288, 291
- L**
- Layer 165
 Leertaste 97
 left 55
 Library 196
 Licht 226
 Lichteinstellungen 228
 Light 226
 Lighting 223
 Light-Mode 18
 linearVelocity 308
 Linecast 334
 LineRenderer 339
 localScale 375
 Lock View 237
 Log 130
 LookAt 366
- M**
- Material 86
 Mathf.Abs 330
 Mesh 159
 Mesh Collider 186
 Mesh Renderer 324
 Methode 50
 MonoBehaviour 50
 MonoDevelop 200
 Mouse X 128
 Mouse Y 181
 Move 84
 moveDirection 201w
- MovePosition 58
 MoveVector 95
- N**
- Navigation
 - not walkable 304
 - vorbereiten 283
 NavigationMesh 283, 388
 Navigator
 - enabled 293
 - stoppen 295
 NavMesh
 - Agent 288
 - Obstacle 305
 - Surface 283
 Nebel 251
 Neues Objekt 16, 26, 72
 Neues Projekt 12
- O**
- Objekt
 - Container 211
 - deaktivieren 282
 - drehen 32
 - duplizieren 136, 188
 - Ebene 72
 - entfernen 282
 - erzeugen 83
 - Größe 32
 - Kugel 16
 - leer 186, 210
 - Licht 20
 - neu 16, 26, 72
 - Quader 26
 - Standardmaße 149
 - umbenennen 39
 - vereinbaren 83
 - verschieben 27
 - Zentrieren 237
 Oder-Operator 201
 Offset 205
 On 114

- OnCollisionEnter 307
 - OnCollisionExit 307
 - OnCollisionStay 307
 - OnControllerColliderHit
 - 200
 - OnTriggerEnter 114, 238, 281
 - OnTriggerExit 114, 238, 281
 - OnTriggerStay 246
 - Opacity 167
 - Operator && 201
 - Operator || 201
 - Ordner
 - erstellen 72
 - Ordner erstellen 44
 - Orthographic 38
 - other 307
- P**
- Package 140
 - Package Manager 142, 161
 - Paket. Siehe Package
 - Parameter 50, 314
 - Parent 134, 369
 - ParticleSystem 352
 - Partikel
 - Ausstoß starten 353
 - Ausstoß stoppen 352
 - Bereiche 350
 - Color 348
 - Emission 347
 - Hauptmodul 347
 - Shape 348
 - Textur 351
 - Zufallswerte 350
 - Partikelsystem 343
 - erzeugen 352
 - Performance 176
 - Pfeiltaste 51
 - Pfeiltasten 79, 95
 - Physics 333
 - Plane 122
 - Platformer 108
 - Plattform 391
 - Play 353
 - Player
 - Dying 321
 - PlayerAudio 356
 - PlayerControl 240, 261
 - PlayerHealth 372, 374
 - PlaySound 356
 - Point Light 226
 - position 59
 - Position 27
 - Child 134
 - Prefab 143, 193, 269
 - erzeugen 209, 217
 - importieren 221
 - private 84
 - Project Settings 93
 - Projekt
 - 3D neu 119
 - Asset-Importe 195
 - entschlacken 196
 - konvertieren 405
 - neu 12
 - umbenennen 336
 - Prototyp 210
 - public 84
 - Punktoperator 53
- Q**
- Quaternion.Euler 321
 - Quelltext 49
 - Quit 395
- R**
- Random 381
 - Range 381
 - Raycast
 - Direction 335
 - Distance 335
 - Position 334
 - sichtbar 340
 - RaycastHit 336
 - Raycasting 333
 - Rechenoperator 54
 - Rect Transform 369
 - Refresh 390
 - remainingDistance 295
 - Renderer
 - Color 114
 - Sprite 99
 - Rendern 88, 99, 251
 - RenderSettings 251
 - return 106, 201, 322
 - right 55
 - Rigidbody 23, 52, 123, 196
 - Eigenschaften 199
 - Rotate 126
 - Rotation 32, 181
 - Rückgängig 155, 176
- S**
- Scale 28, 32, 122
 - Scene
 - speichern 73
 - Schlüssel-Nummer 318, 328
 - Schlüsselwort 50
 - Schwimm-Modus 260
 - Screen 373
 - Script 43
 - doppelt 387
 - Editor 50
 - erstellen 45
 - Sprache 46
 - Second Person 121
 - SendMessage 339, 358
 - SetActive 282
 - SetBool 318
 - SetDestination 291
 - SetPosition 342
 - SetZero 260
 - Shader 88
 - Shape 348
 - Shuriken 343

- Sin 364
 - Skalierung 32
 - Skybox 369
 - Slope Limit 158, 230
 - Sound 353
 - SphereCast 334
 - Sphere Collider 186
 - SphereControl 49
 - Spiel
 - erzeugen 391
 - Plattform 391
 - Szene 44
 - Spiele-Engine 9
 - Spot Light 226
 - Sprite 73
 - Bild 99
 - Sprite Renderer 99, 100
 - sqrMagnitude 308
 - Standard Assets 145
 - Start 50
 - Startwerte 85
 - State 312
 - Machine 312
 - Steigung 158, 229
 - Step Offset 232
 - Stop 352
 - stoppingDistance 295
 - Strahl
 - Dicke 340
 - String 82
 - StringToHash 318
 - Szene
 - wiederherstellen 91
 - Szene zoomen 155
- T**
- Tauch-Modus 260
 - Terrain
 - Baum 170
 - erweitern 151
 - erzeugen 147
 - Gras 174
 - Grenzen 184
 - Höhe 151
 - Höhle 159
 - Layer 165
 - Maße 150
 - Mitte 157
 - Plateau 152, 153
 - Settings 176
 - Size und Opacity 154
 - Smooth 153
 - Stamp 154
 - Textur 164
 - Textur hinzufügen 166
 - Werkzeug 150
 - Wind 177
 - Terrain Assets 162
 - Terrain Collider 186
 - Terrain Pack 161
 - TextMeshPro 385
 - Textur 86
 - auftragen 167
 - einsetzen 116
 - importieren 115
 - Paint 164
 - Terrain 164
 - Tiling 205
 - Third Person 121
 - Tiling 205, 225
 - Time 60, 357
 - Tools
 - wiederherstellen 37
 - Touch 95
 - Transform 32, 78, 125
 - Transformation 36
 - TransformDirection 129, 182
 - Transitions 315
 - Translate 78, 125
 - Transparenz 243
 - Tree Assets 164
 - Tree-Objekt 169
 - Treppe 231
 - Trigger 113
 - Klettern 236
 - sichtbar 236
 - unsichtbar 235, 249
 - Wasser 249
 - true 139
- U**
- Umbenennen 336
 - Und-Operator 201
 - Unity
 - beenden 40
 - starten 11
 - Unity-AI 291
 - UnityEngine 49
 - Unity Hub 397
 - Unity Links 396
 - Unity-TMP 385
 - Unity-UI 375
 - Unterwasser 251
 - up 55
 - Update 50
 - Ursprung 34
 - using 49
- V**
- Variable 82
 - Inspector-Fenster 85
 - Vector3 54, 124, 200
 - Vektor 54, 95, 125
 - velocity 358
 - Velocity 308
 - Verbindungsoperator 53
 - Vereinbarung 51
 - Vergleichsoperator 102
 - Verknüpfung 12
 - Verknüpfungsfehler 387
 - Verknüpfungsoperator 201
 - Verneinungsoperator 240
 - Visual Studio 47

W

Warnung 409
Wasser 193
 Höhe ermitteln 259
Wasser-Trigger 249
Waten-Modus 260
WaterControl 251
Water-Objekt 247
Werkzeug
 Baum 168
 Detail 174

Details 174

Gestaltung 151
Wheel Collider 186
Windeinstellungen 177
Wind Zone 189

X

x-Achse 34

Y

y-Achse 34

Z

z-Achse 35
Zeitverzögerung 328
Zentrierung 157
zero 95
Zoom 28, 155, 192
Zufallswerte 350
Zuweisung 52
Zuweisungsoperator 83