Hans-Georg Schumann

3D-Spiele programmieren mit Unity

Ganz einfach ohne Vorkenntnisse

mitp

3D-Spiele programmieren mit Unity

Ganz einfach ohne Vorkenntnisse

Hans-Georg Schumann

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über https://portal.dnb.de/opac.htm abrufbar.

ISBN 978-3-7475-0924-1
1. Auflage 2025

www.mitp.de
E-Mail: mitp-verlag@lila-logistik.com
Telefon: +49 7953 / 7189 - 079
Telefax: +49 7953 / 7189 - 082

Lektorat: Katja Völpel
Sprachkorrektorat: Jürgen Dubau
Covergestaltung: Christian Kalkert
Bildnachweis: © Damian Sobczyk / Photocreo Bednarek / stock.adobe.com
Satz: Petra Kleinwegen

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

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

Einleitung

E.1 Spielentwicklung

Früher hat man ein Spiel komplett »von Hand« programmiert, also das gesamte Spiel in einer Programmiersprache. Anfangs hatte man nicht viele Möglichkeiten, deshalb begann alles mit viel Text. Grafik gab es eigentlich gar nicht, aber man versuchte, mit Textsymbolen etwas zu erzeugen, was dann wie ein einfacher Gegenstand oder gar eine Figur aussehen konnte.

Mit der Zeit entwickelten immer mehr findige Programmierer Module, mit denen sich dann auch komplexere Dinge erzeugen ließen. Und Spiele wurden immer ansehnlicher. Als dann die erste wirkliche Grafik möglich war, gab es alsbald auch Module, mit deren Hilfe sich immer besser aussehende Figuren und Gegenstände darstellen ließen, dann auch attraktivere Hintergründe.

Und irgendwann gab es dann eine umfangreiche Sammlung solcher Module, die als Spiele-Engine oder Game-Engine bezeichnet wurde. Weil Computer- und Konsolen-Spiele immer beliebter wurden, traten natürlich eine ganze Reihe von Herstellern auf den Plan. Die verwendeten oft ihre eigene Game-Engine. Mit der Zeit wurden dann auch Entwicklungssysteme für Spiele angeboten, die für jeden zugänglich waren – teilweise kostenlos, zum Teil kostenpflichtig.

E.2 Und was ist Unity?

Unity Technologies ist eine Firma, die ein umfangreiches System anbietet, mit dem man nicht nur Spiele programmieren kann. Die Game-Engine, mit der wir hier arbeiten, ist eine kostenlose Version. Zahlen muss man erst, wenn man viel Geld mit Spielen verdient (wieviel aktuell, das erfahren Sie über die Unity-Website).

Unity kann mit physikalischen Gesetzen umgehen, damit die Spielwelt mit ihren Figuren und Ereignissen in der jeweiligen Spielumgebung möglichst echt wirkt. Und dazu muss diese Engine komplexe grafische Effekte beherrschen, um für eine hervorragende Optik zu sorgen.

Mit Unity haben Sie nicht nur ein vollwertiges System für Spielentwicklung, das so vielfältige Möglichkeiten bietet, deren komplette Beschreibung nie in dieses Buch passen würde. Aber die wichtigsten lernen Sie hier kennen.

Mit dem visuellen Editor lässt sich ein Spiel bequem erstellen – und das nicht nur für Windows, sondern auch für andere Plattformen wie z.B. Linux, Android oder iOS.

Welche Unity-Version?

Ich selbst kenne Unity seit Version 5 (von 2016). Die danach folgenden Versionen wurden nach den Jahren benannt, in denen sie erschienen. Die aktuelle, in diesem Buch verwendete ist Version 6, was eigentlich nach der alten Zählung Unity 2024 wäre.

E.3 Voraussetzungen

Sie brauchen einen Computer mit folgenden Eigenschaften:

- 64 Bit Windows 10 oder 11, am besten die jeweils neueste Version
- einen Prozessor vom Typ Intel Core-i oder AMD Ryzen Multi-Core, empfohlen Intel i5 oder schneller
- eine nicht zu schwache Grafikkarte, die mit Microsoft DX10/11/12 funktioniert
- RAM mit mindestens 4 GB RAM, empfohlen 8 oder 16 GB
- freien Speicherplatz auf dem Datenträger von mehr als 20 GB
- eine empfohlene Bildschirmauflösung mit 1920 x 1080

Grundsätzlich kann ein Mehr an Leistung und Platz nicht schaden.

Wie Unity heruntergeladen und installiert wird, erfahren Sie im Anhang. Die Projekte zum Buch können Sie in einem Paket von der Verlagsseite herunterladen:

https://www.mitp.de/0923

Beachten Sie dabei, dass die Projekte wahrscheinlich konvertiert werden müssen, denn von Unity erscheinen häufig neue Updates. Die Konvertierung erledigt Unity aber automatisch.

Kapitel 1

Das erste Projekt

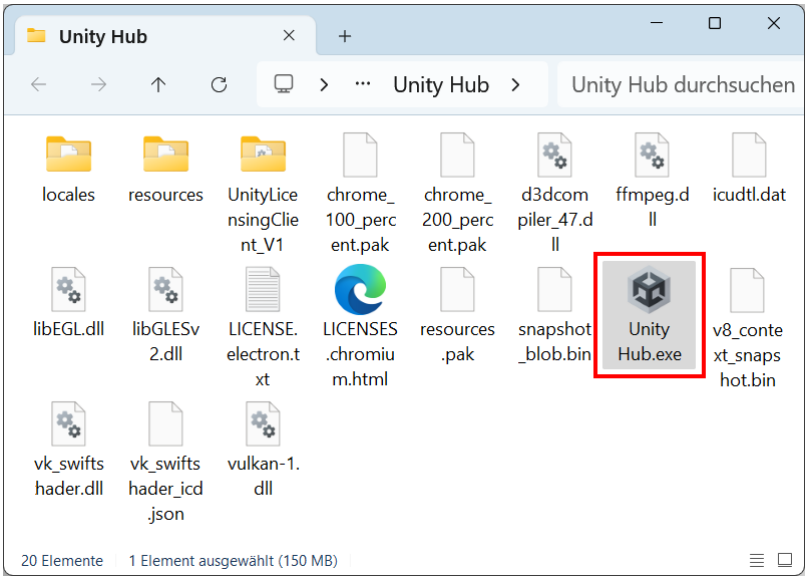
Bevor Sie Ihr erstes Spiel erstellen können, müssen Sie sich noch ein bisschen gedulden. Erst einmal machen Sie sich etwas mit der »Maschine« vertraut, mit der Sie später ein Werk »zaubern« wollen, das sich sehen und spielen lässt. Schon hier beginnen wir mit einem Projekt. Und wir spielen auch schon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



2. 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

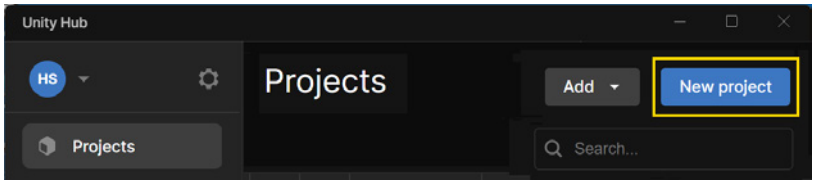
Weil wir Unity ja sehr oft starten werden, empfehle ich hier, eine Verknüpfung auf dem Desktop anzulegen:

- 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.

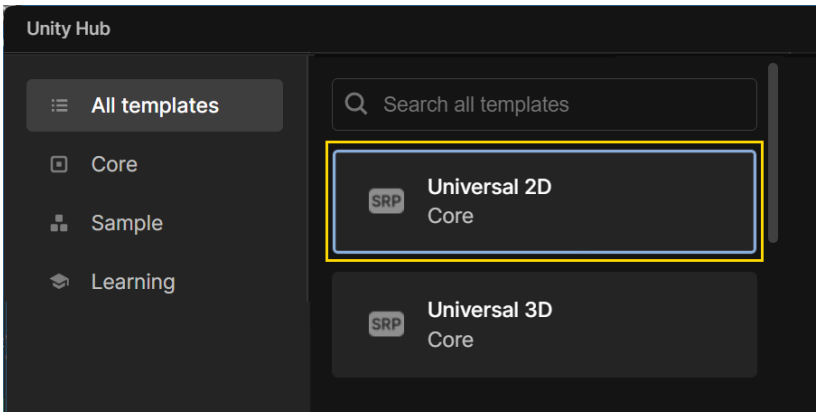
Es ist sinnvoll, das neue Symbol auf dem Desktop umzubenennen, z.B. von UNITY HUB.EXE – VERKNÜPFUNG in einfach nur UNITY.

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:



3. Klicken Sie dazu auf NEW PROJECT.

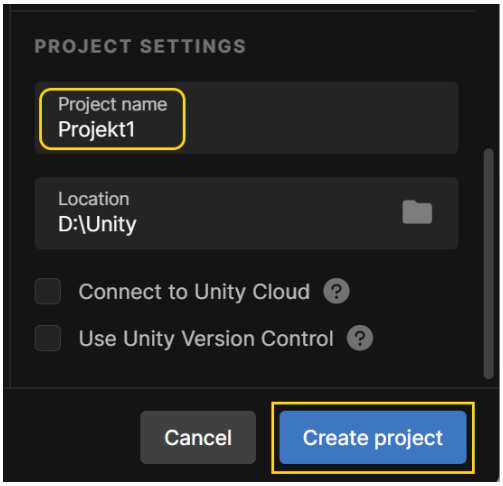


4. Wählen Sie im neuen Dialogfeld die Einstellung Universal 2D.

Mit 3D beschäftigen wir uns später noch ausführlich.

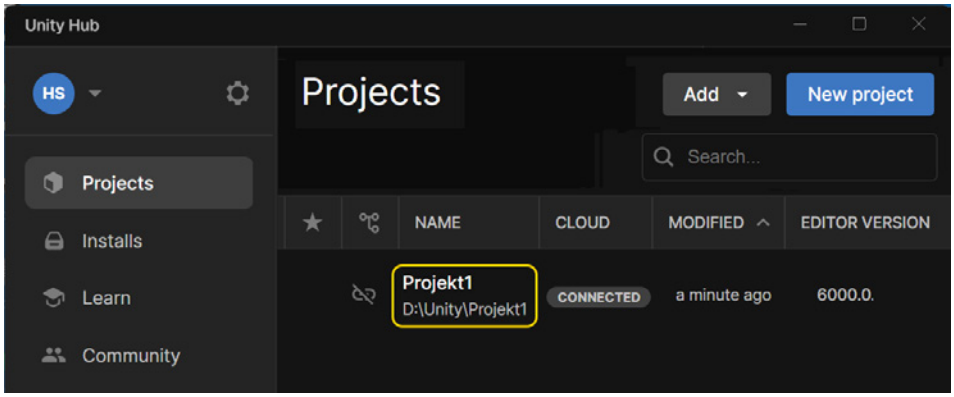
5. Geben Sie dann im Feld für Project name einen Namen für Ihr neues Projekt ein. Bei Location sollte der Ordner stehen, in dem das Projekt untergebracht werden soll (wenn Sie nichts eingeben, schafft sich Unity seinen eigenen Ordner für Ihre Spielprojekte.)

Ich benutze einen Ordner Unity und nenne mein erstes Projekt schlicht und einfach Projekt1.

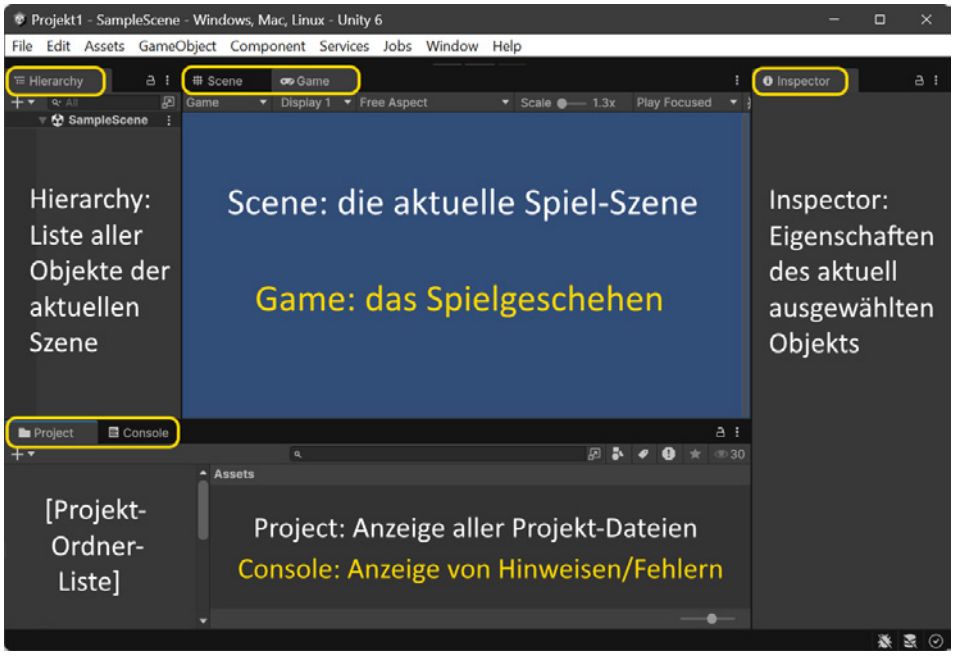


6. Klicken Sie zum Abschluss auf Create project.

Es dauert nun eine Weile, bis Ihr Projekt in der Liste unter Projects auftaucht.



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

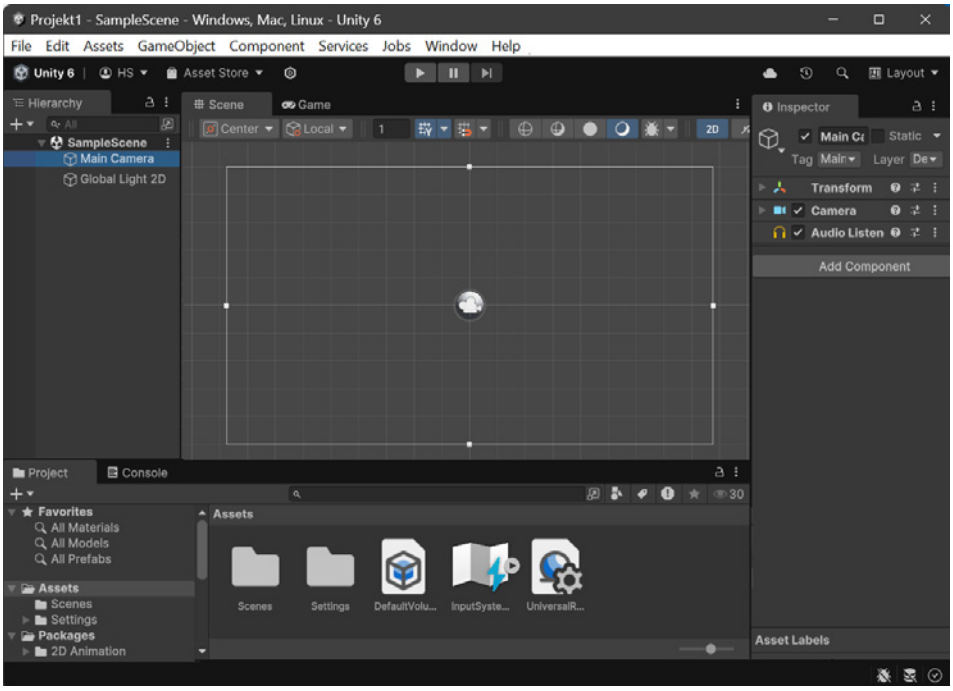


- Im GAME-Fenster ist es erst einmal leer. Da sehen Sie später Ihr Spiel in Echtzeit ablaufen, wenn Sie es durch einen der darüberliegenden Buttons gestartet haben.
- Dahinter liegt das SCENE-Fenster. Und es gibt auch schon zwei Objekte: Kamera und Licht. Doch für ein Spiel brauchen wir dann noch mindestens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



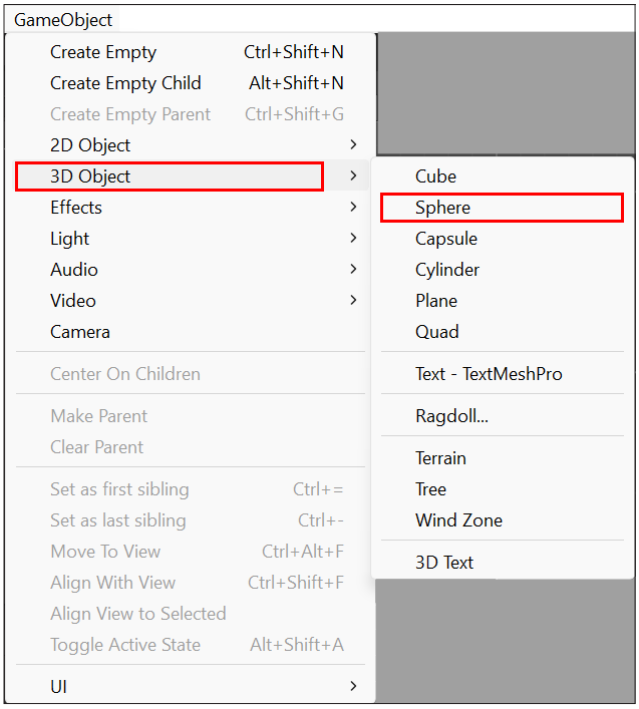
Wenn Sie im Hierarchy-Fenster auf Main Camera klicken, zeigt der Inspector plötzlich eine ganze Menge an (ändern sollten Sie aber daran nichts).

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

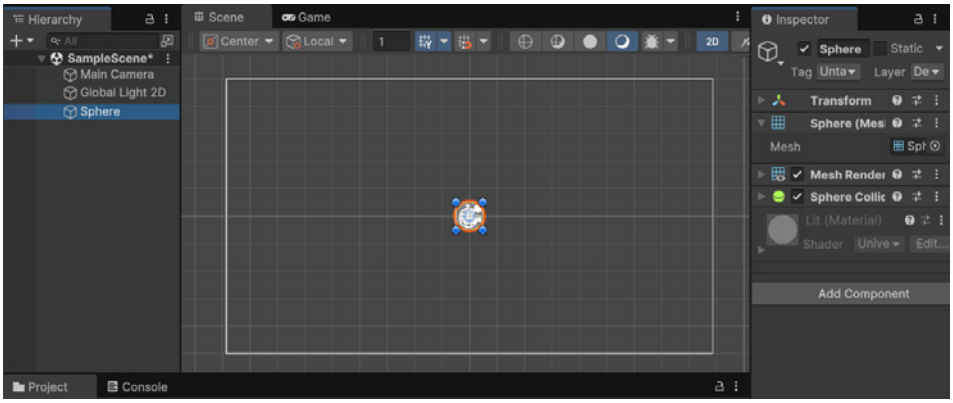
1.2 Ein Objekt zum Spielen

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).



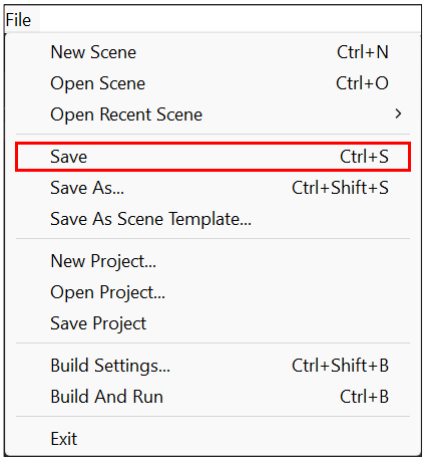
Anschließend taucht im Scene-Fenster etwas auf, das bei genauerem Hinsehen wie ein Kreis aussieht – aber irgendwie auch recht mickrig. Außerdem zeigt der Inspector zahlreiche Informationen über unser neues Spiel-Objekt.



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.

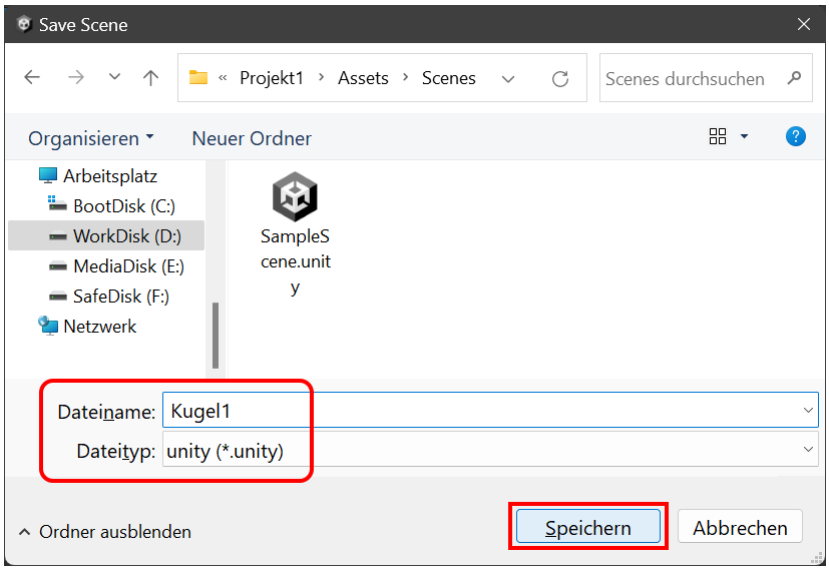
Nun ist es an der Zeit, die ganze Szene schon einmal zu speichern.

2. Klicken Sie auf File und Save.

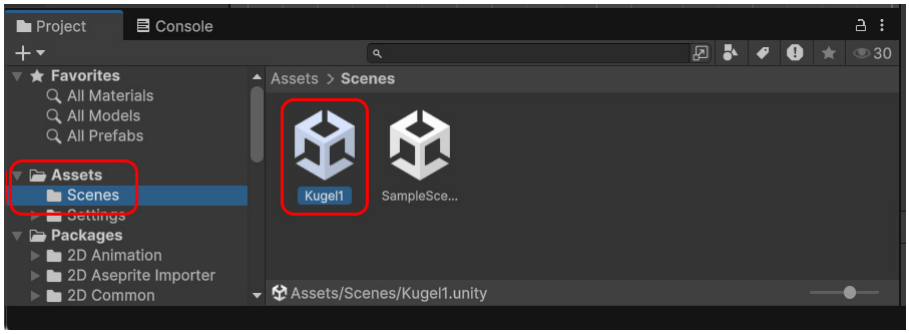


SampleScene

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.



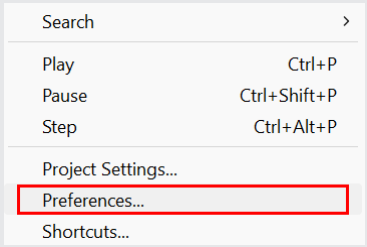
Wenn Sie anschließend im Project-Fenster auf Assets klicken, sehen Sie den Ordner Scenes, darin befindet sich das Symbol für die Szenen-Dateien.



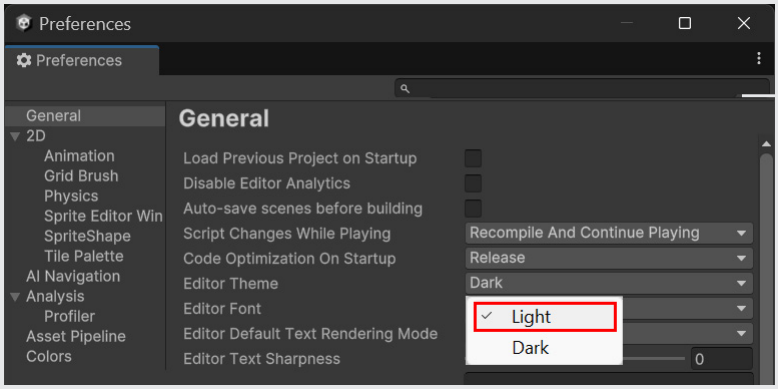
Dark oder Light

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.

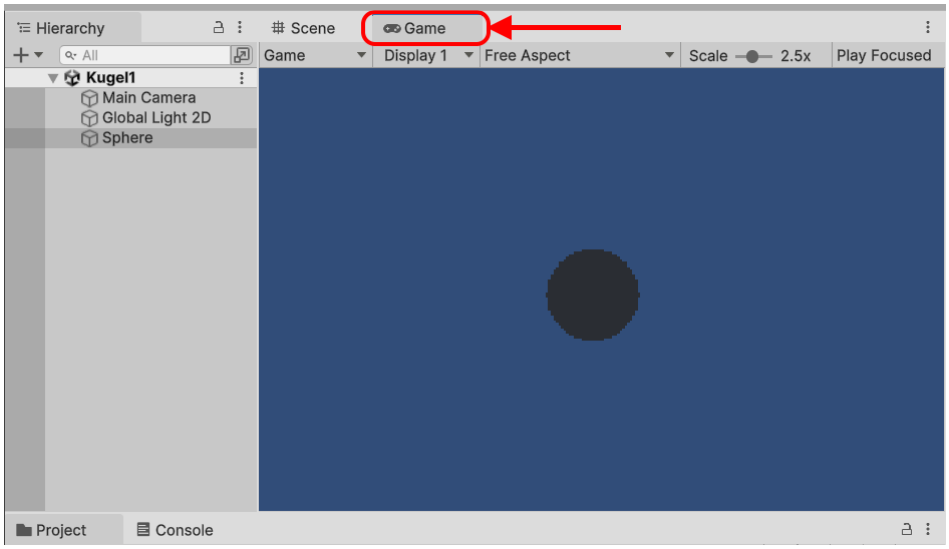


Anschließend ist alles um einiges heller. Wenn es Ihnen gefällt, lassen Sie es so. Oder Sie kehren zurück zum Dark-Mode.

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.

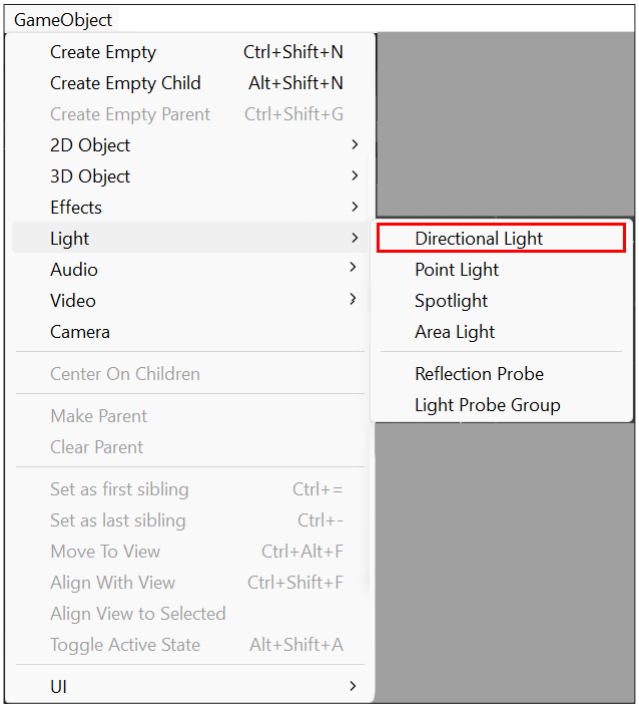
Und nun schauen wir uns die ganze Szene einmal genauer an, und zwar in einem anderen Fenster.

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

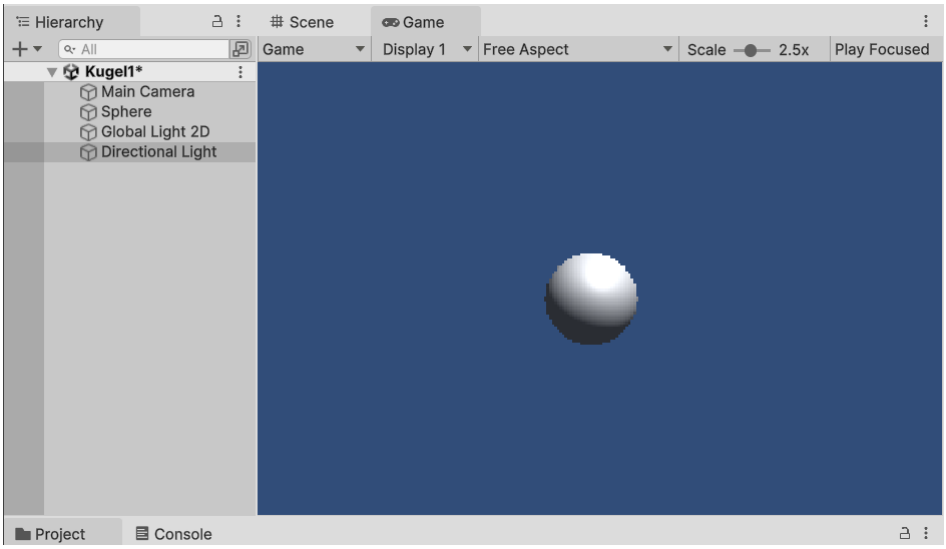


Im Game-Fenster können Sie nun etwas sehen. Naja, wie eine Kugel schaut dieser schwarze Kreis (noch) nicht aus. Aber vielleicht lässt sich daran etwas ändern.

2. Klicken Sie im Hauptmenü auf GameObject und dann auf Light. Im Zusatzmenü wählen Sie den Eintrag Directional Light.

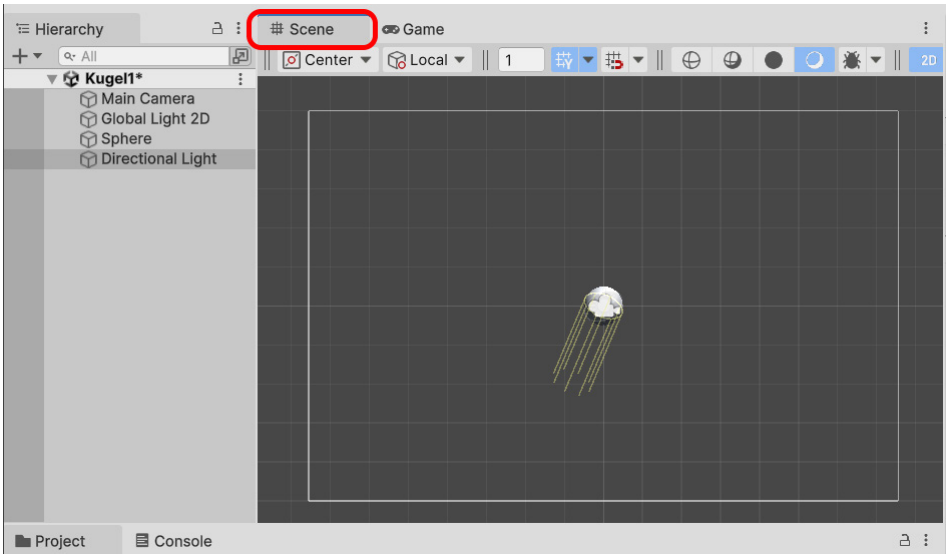


Kurz darauf sehen Sie die Kugel in einem anderen Licht. Erkennt man das nur im Game-Fenster?



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.



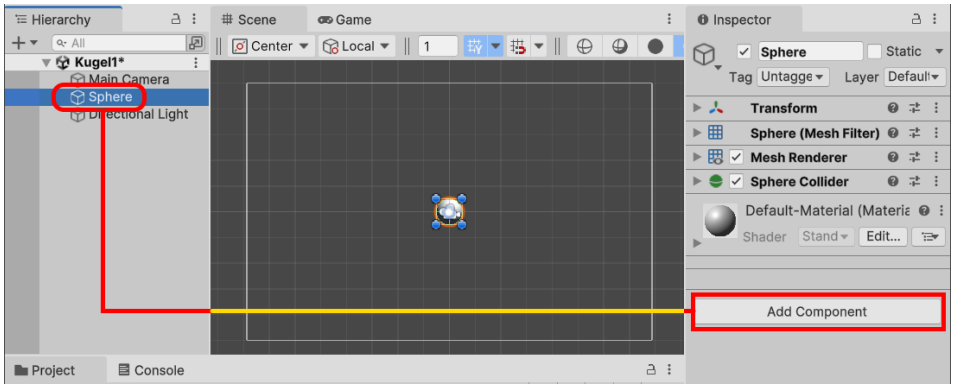
Nun gibt es ein weiteres Objekt, und es liegt ebenso wie die Kugel in der Fenstermitte.

Licht-Objekte

Wenn Sie wollen, können Sie Global Light auch entfernen: Markieren Sie den Eintrag und drücken Sie die Taste Entf.

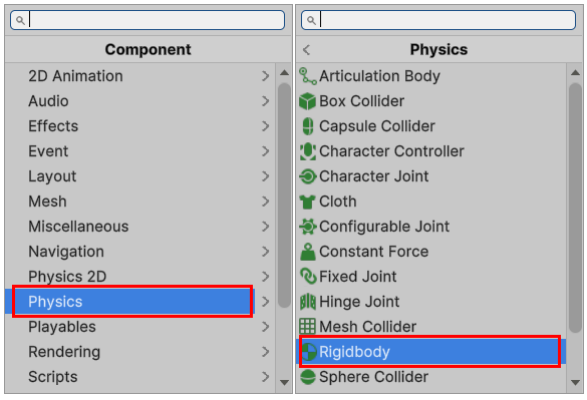
Toll wäre es, wenn sich die Kugel bewegen würde. Doch wie kriegen wir sie dazu? Zuerst einmal sollten wir diesem »Ding« physikalische Eigenschaften geben. Dass das Objekt aussieht wie eine Kugel, heißt noch nicht, dass es sich auch wie eine Kugel aus einem bestimmten Material verhält.

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.



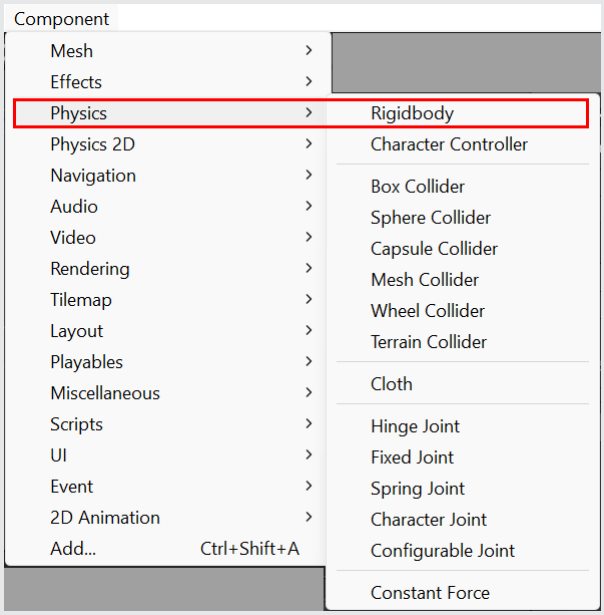
Ein kleines Kontextmenü öffnet sich.

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.

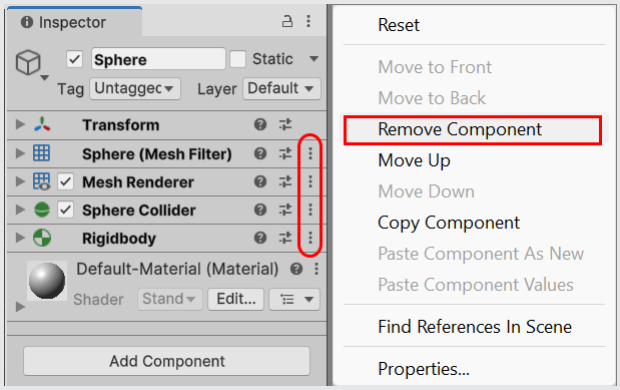


Damit bekommt die Kugel (Sphere) einige physikalische Eigenschaften wie *Masse* (unsere Kugel wiegt also auf einmal etwas) und *Gravitation* (sie wird vom Boden angezogen, würde also aus der Luft herunter auf den Boden fallen). Auch Kollisionen mit anderen Objekten und ihre Folgen sind nun möglich (ich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

Komponente wieder entfernen?

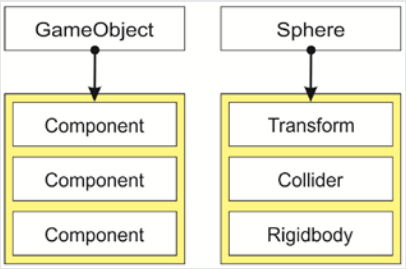
Falls Sie aus Versehen eine falsche Komponente hinzugefügt haben: Wie werden Sie diese wieder los? Schauen Sie im Inspector-Fenster mal genauer hin. Hinter jedem Komponenten-Namen sind ganz rechts drei kleine Pünktchen.

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.

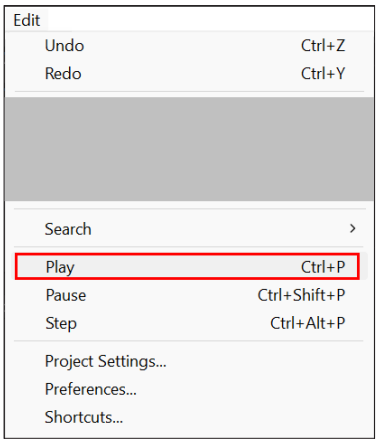


Die Komponente Transform hat jedes Spiel-Objekt »von Geburt an«. Weitere Komponenten lassen sich (fast) beliebig hinzufügen (aber auch wieder entfernen).

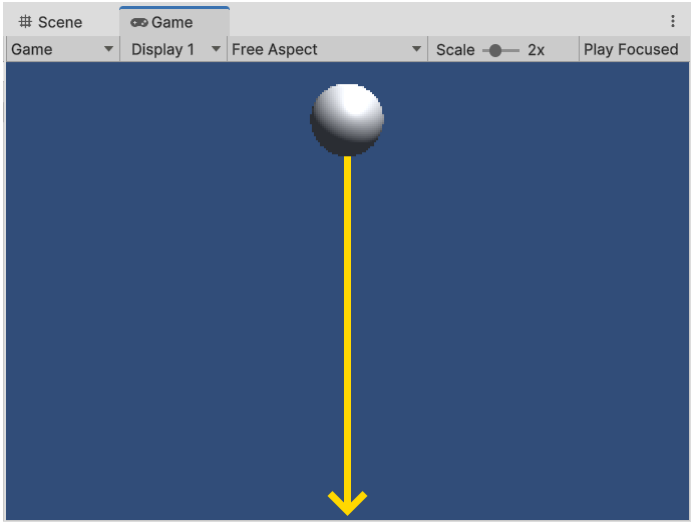
Wir sollten das Ganze schon einmal ausprobieren.

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.

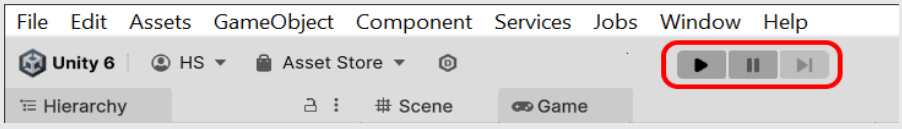


Im ersten Moment passiert anscheinend nichts, dann auf einmal fällt die Kugel und verschwindet aus dem Spielfeld.



Start und Stopp

Start und Stopp eines Unity-Spiels können Sie auch über Buttons steuern. Ganz oben, direkt unter der Menüzeile, 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.

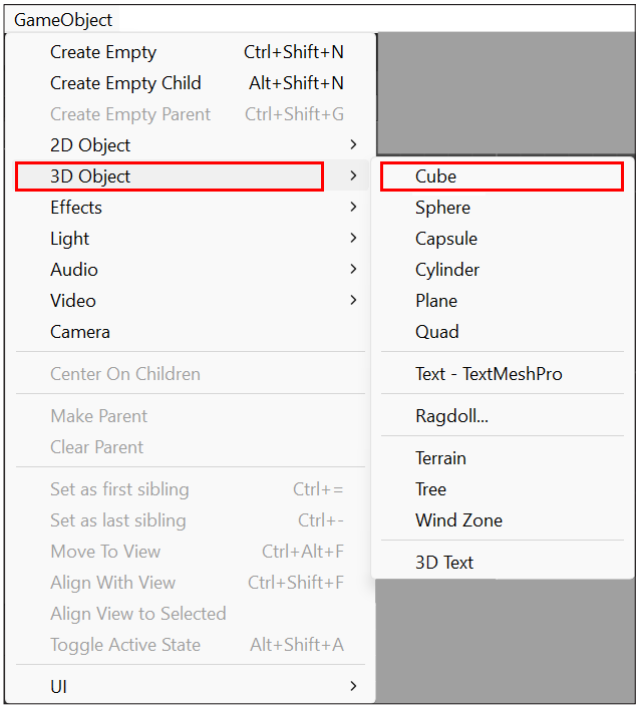
1.3 Gravitation und Kollision

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.

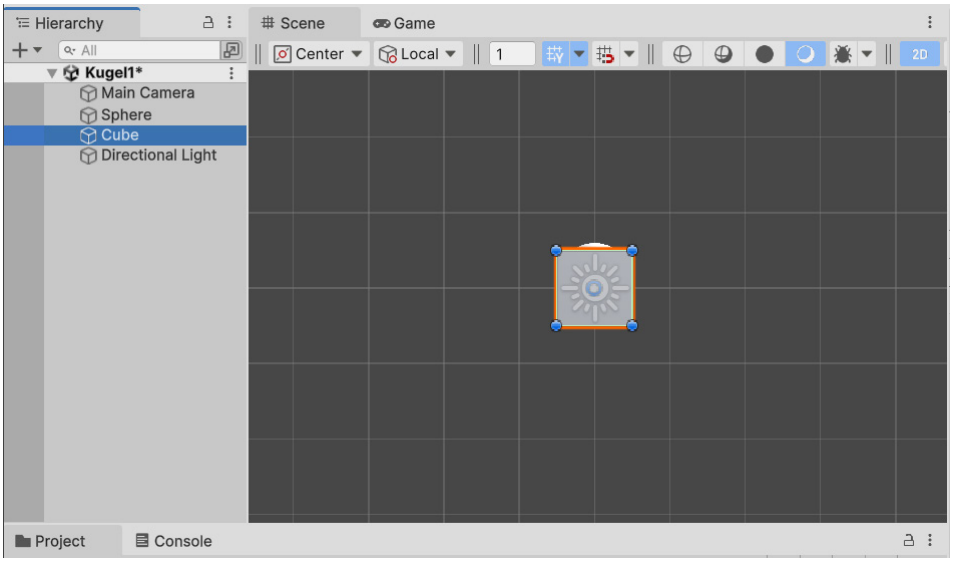
Andererseits kann die Gravitation einem beim Spielen nützlich sein. Denken Sie z.B. an ein Jump & Run-Game. Da geht es ja um Figuren, die springen und rennen, aber immer wieder irgendwo landen (und dazu brauchen sie die »Erdanziehungskraft«).

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.



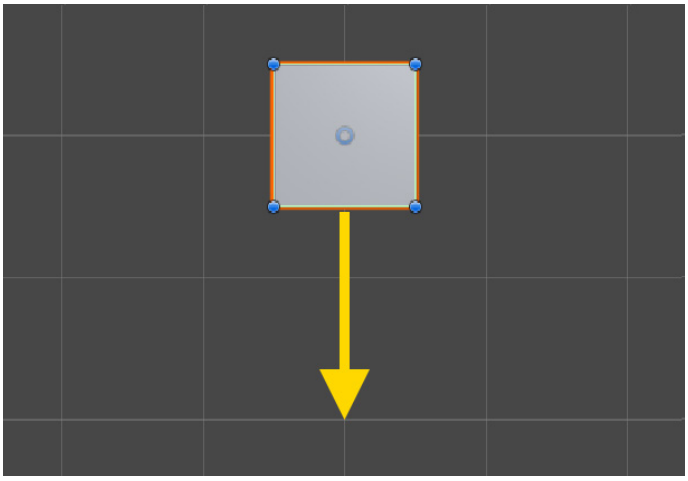
Im Scene-Fenster hat sich nun über den Kreis so etwas wie ein Quadrat gelegt. Auch hier zeigt der Inspector zahlreiche Informationen über das neue Spiel-Objekt.



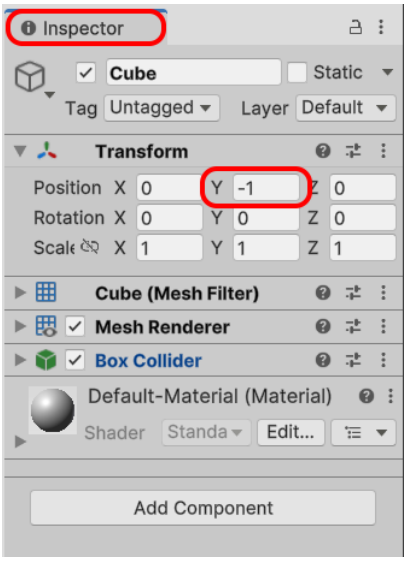
2. Vielleicht ist es sinnvoll, die Szene erst einmal wieder speichern (mit Klick auf File und Save).

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:

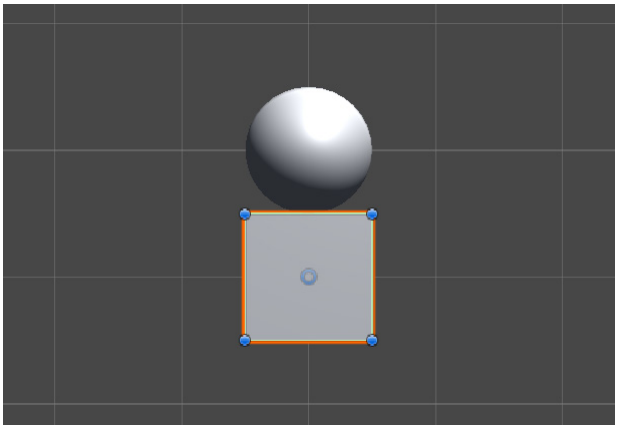
- Man packt das Objekt im Scene-Fenster mit der Maus und zieht es nach unten (das Ganze geht natürlich auch in andere Richtungen).



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



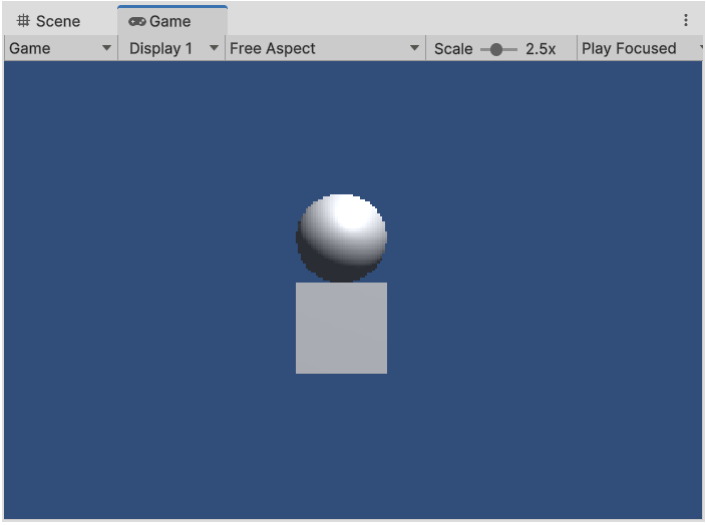
3. Sorgen Sie dafür, dass der Quader direkt unter der Kugel liegt.



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).

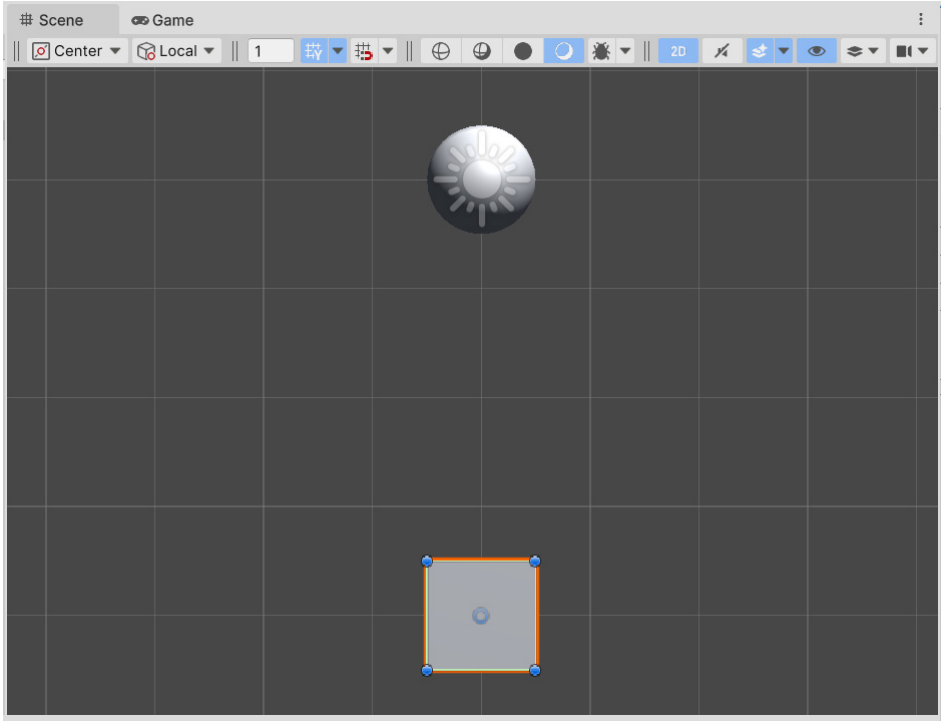
Zoom

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.



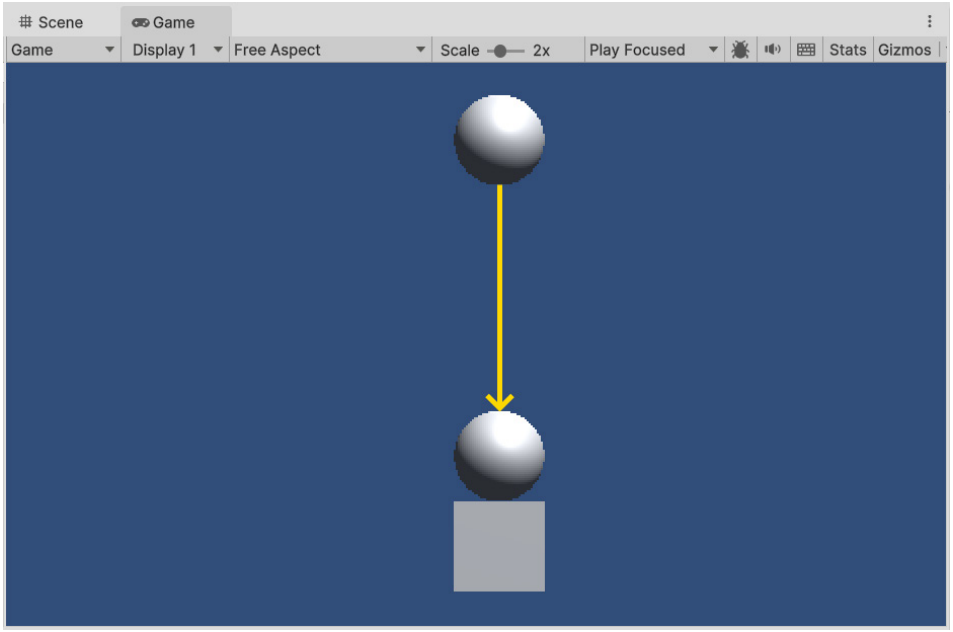
Bei mir steht im Inspector-Fenster hinter Position der y-Wert -5. Bei Ihnen kann da natürlich auch etwas anderes stehen.

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.