Hans-Georg Schumann

3D-Spiele programmieren mit Unity

Ganz einfach ohne Vorkenntnisse

mitp

Inhaltsverzeichnis

Kapitel 1

Das erste Projekt

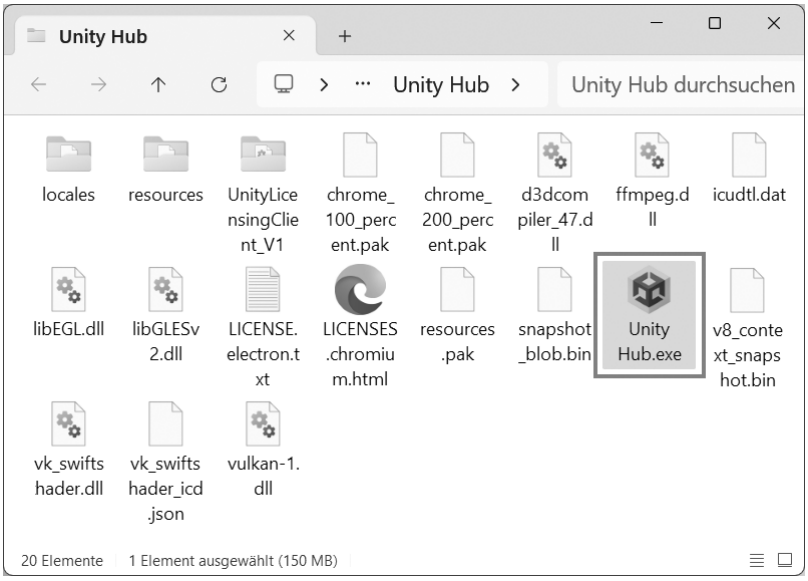
Bevor Sie Ihr erstes Spiel erstellen können, müssen Sie sich noch ein bisschen gedulden. Erst einmal machen Sie sich etwas mit der »Maschine« vertraut, mit der Sie später ein Werk »zaubern« wollen, das sich sehen und spielen lässt. Schon hier beginnen wir mit einem Projekt. Und wir spielen auch schon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

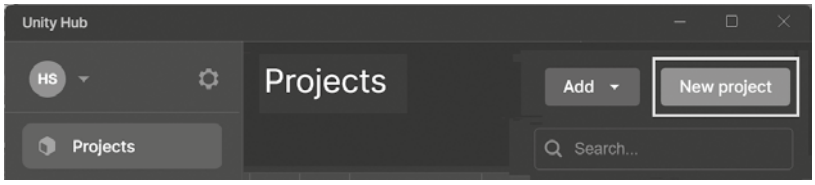
Weil wir Unity ja sehr oft starten werden, empfehle ich hier, eine Verknüpfung auf dem Desktop anzulegen:

- 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.

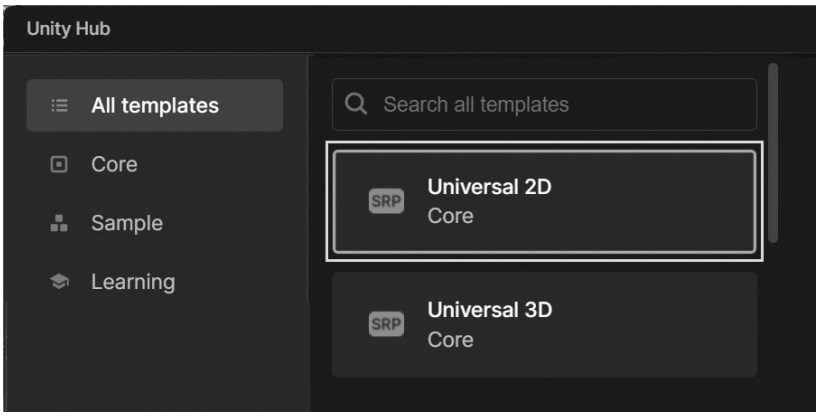
Es ist sinnvoll, das neue Symbol auf dem Desktop umzubenennen, z.B. von unity hub.exe – Verknüpfung in einfach nur Unity.

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.

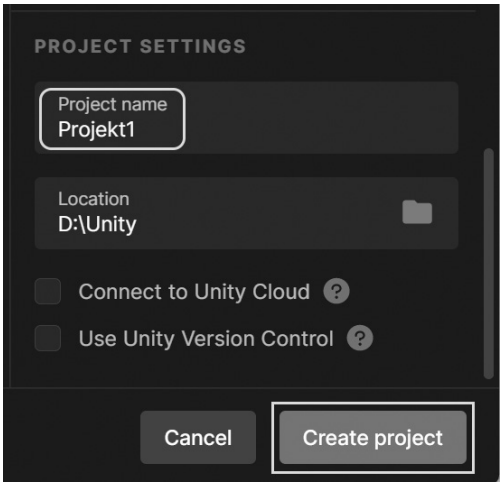


4. Wählen Sie im neuen Dialogfeld die Einstellung Universal 2D.

Mit 3D beschäftigen wir uns später noch ausführlich.

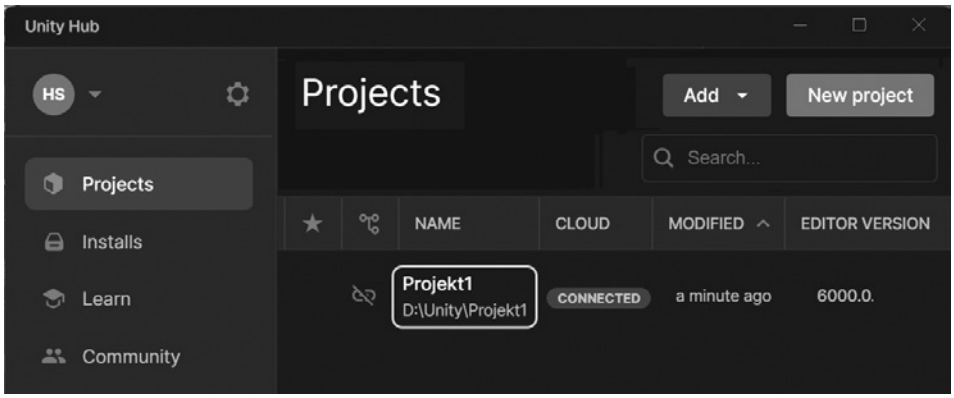
5. Geben Sie dann im Feld für Project name einen Namen für Ihr neues Projekt ein. Bei Location sollte der Ordner stehen, in dem das Projekt untergebracht werden soll (wenn Sie nichts eingeben, schafft sich Unity seinen eigenen Ordner für Ihre Spielprojekte.)

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

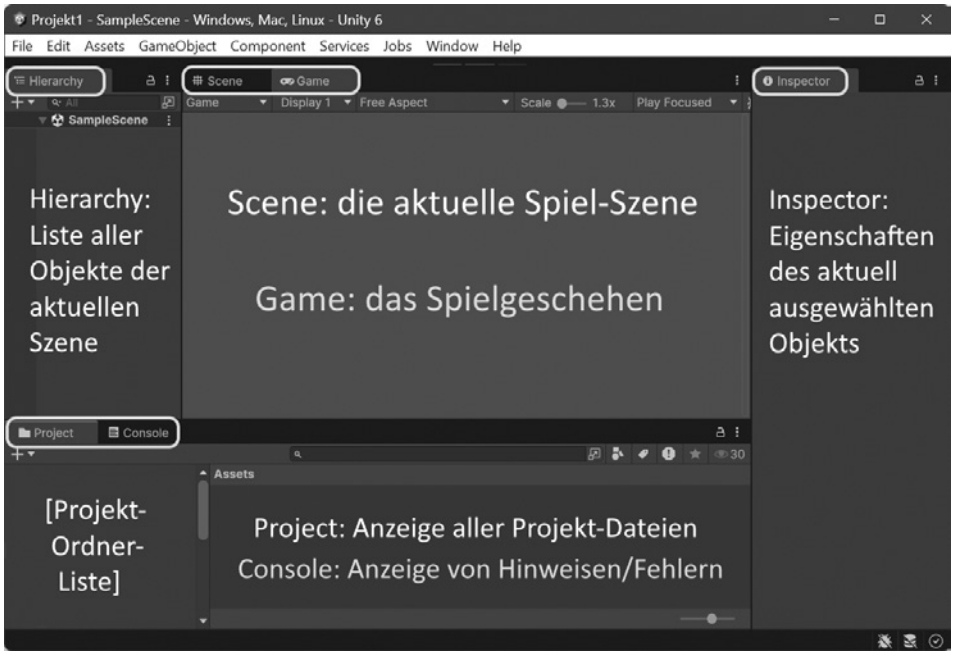


6. Klicken Sie zum Abschluss auf Create project.

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:

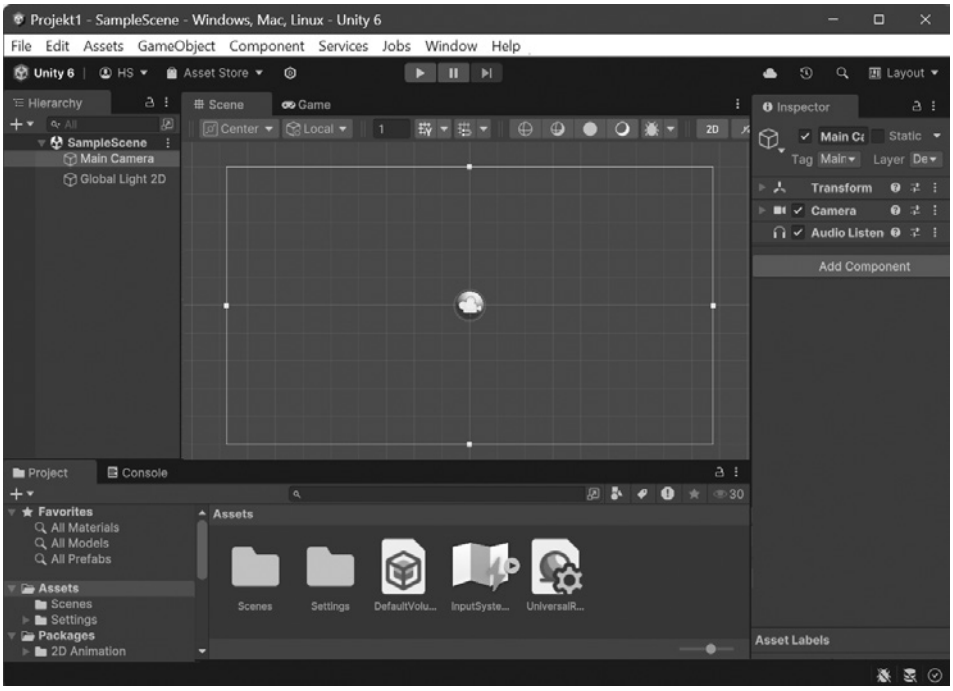


- Im GAME-Fenster ist es erst einmal leer. Da sehen Sie später Ihr Spiel in Echtzeit ablaufen, wenn Sie es durch einen der darüberliegenden Buttons gestartet haben.
- Dahinter liegt das SCENE-Fenster. Und es gibt auch schon zwei Objekte: Kamera und Licht. Doch für ein Spiel brauchen wir dann noch mindestens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



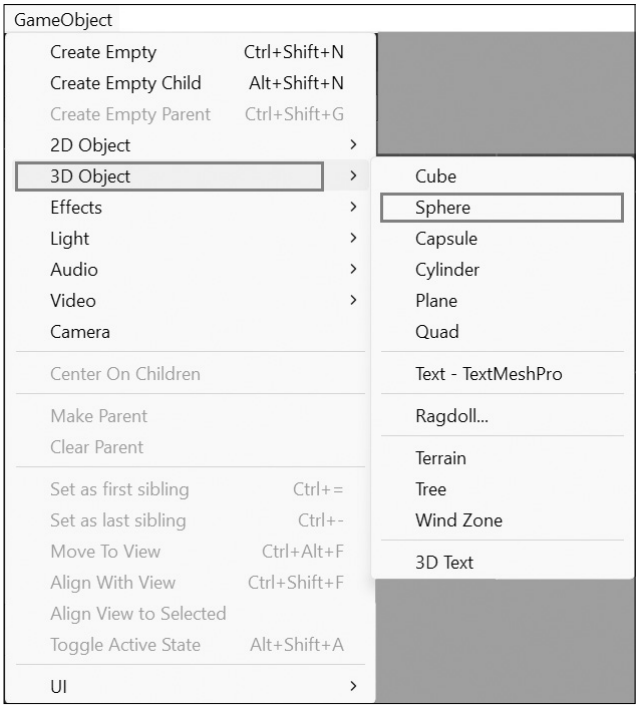
Wenn Sie im Hierarchy-Fenster auf Main Camera klicken, zeigt der Inspector plötzlich eine ganze Menge an (ändern sollten Sie aber daran nichts).

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

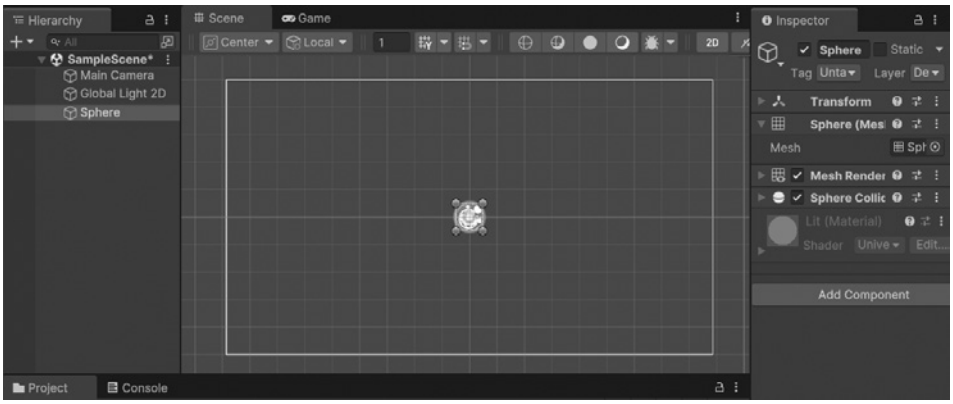
1.2 Ein Objekt zum Spielen

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).



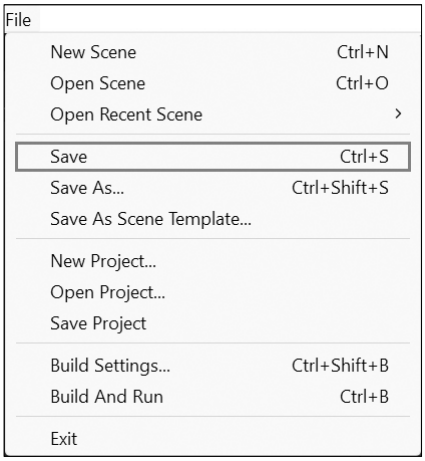
Anschließend taucht im Scene-Fenster etwas auf, das bei genauerem Hinsehen wie ein Kreis aussieht – aber irgendwie auch recht mickrig. Außerdem zeigt der Inspector zahlreiche Informationen über unser neues Spiel-Objekt.



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.

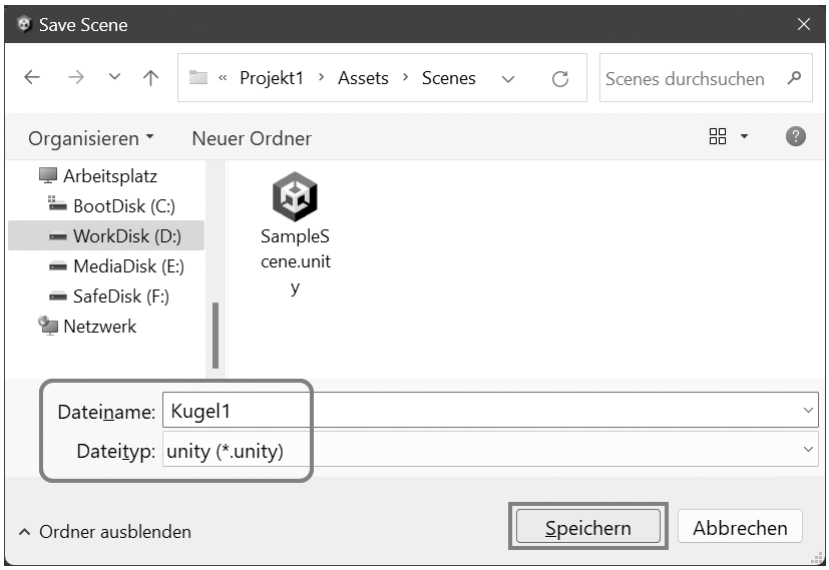
Nun ist es an der Zeit, die ganze Szene schon einmal zu speichern.

2. Klicken Sie auf File und Save.

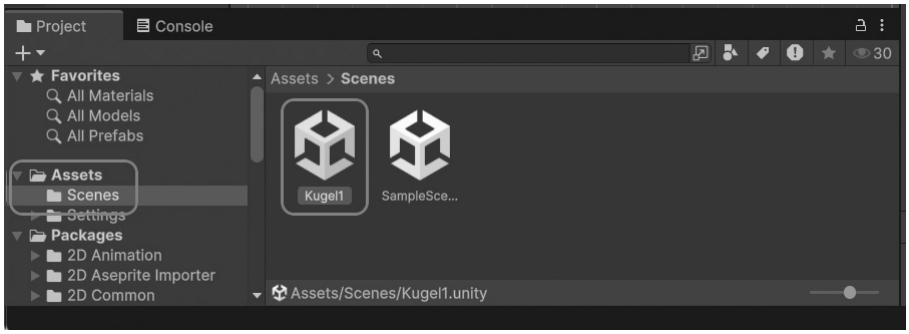


SampleScene

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.



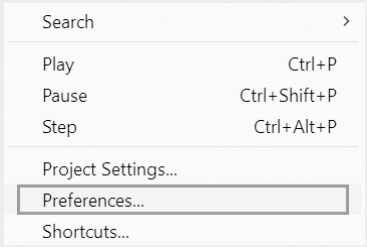
Wenn Sie anschließend im Project-Fenster auf Assets klicken, sehen Sie den Ordner Scenes, darin befindet sich das Symbol für die Szenen-Dateien.



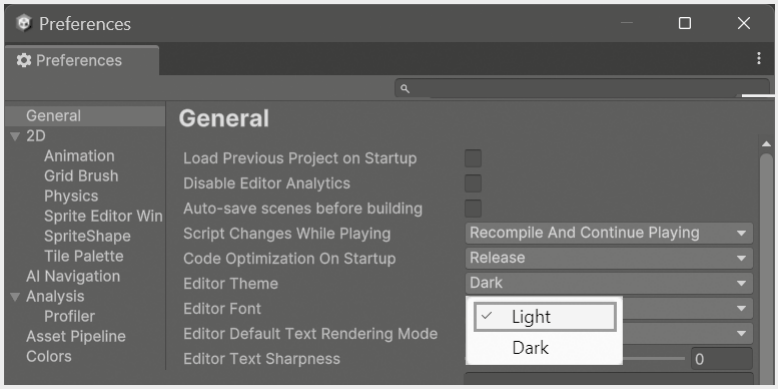
Dark oder Light

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.

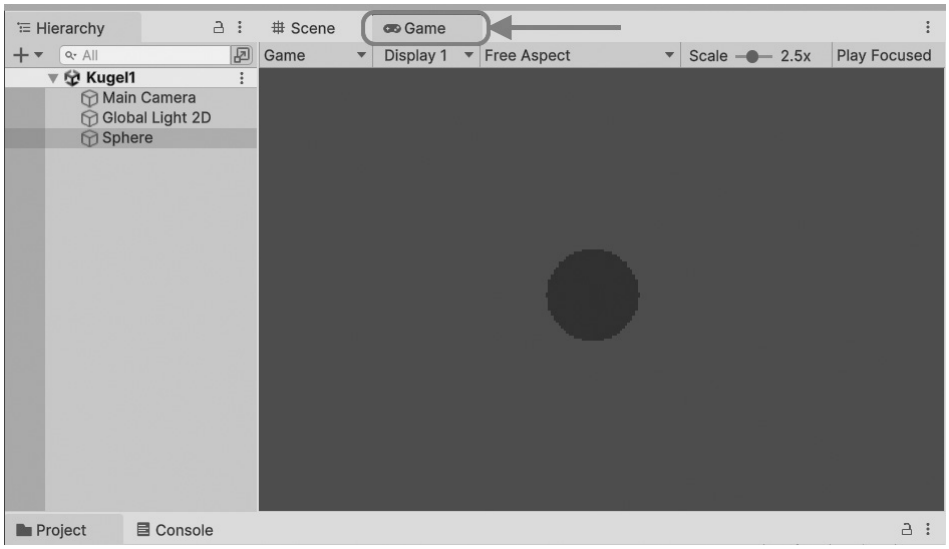


Anschließend ist alles um einiges heller. Wenn es Ihnen gefällt, lassen Sie es so. Oder Sie kehren zurück zum Dark-Mode.

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.

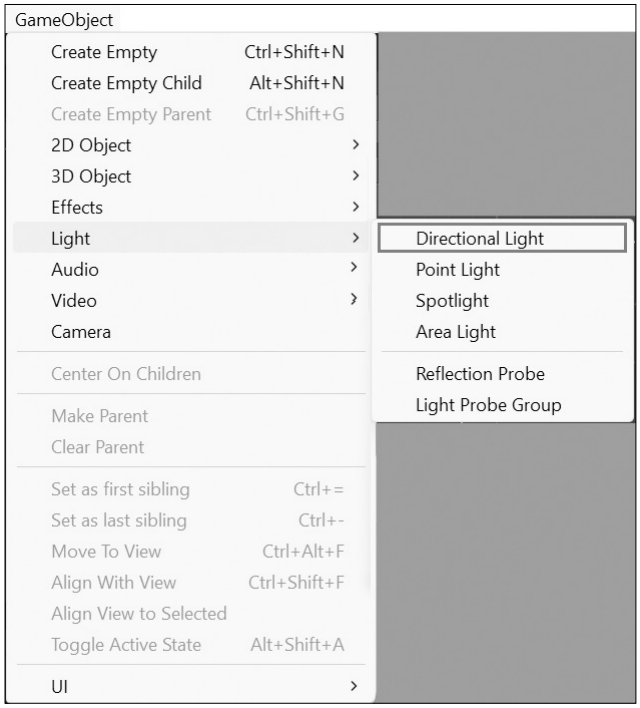
Und nun schauen wir uns die ganze Szene einmal genauer an, und zwar in einem anderen Fenster.

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

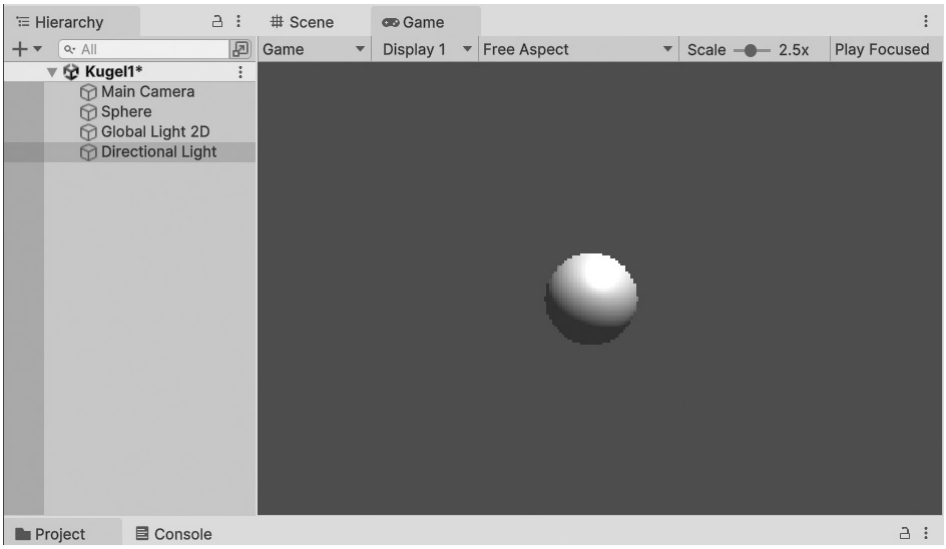


Im GAME-Fenster können Sie nun etwas sehen. Naja, wie eine Kugel schaut dieser schwarze Kreis (noch) nicht aus. Aber vielleicht lässt sich daran etwas ändern.

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

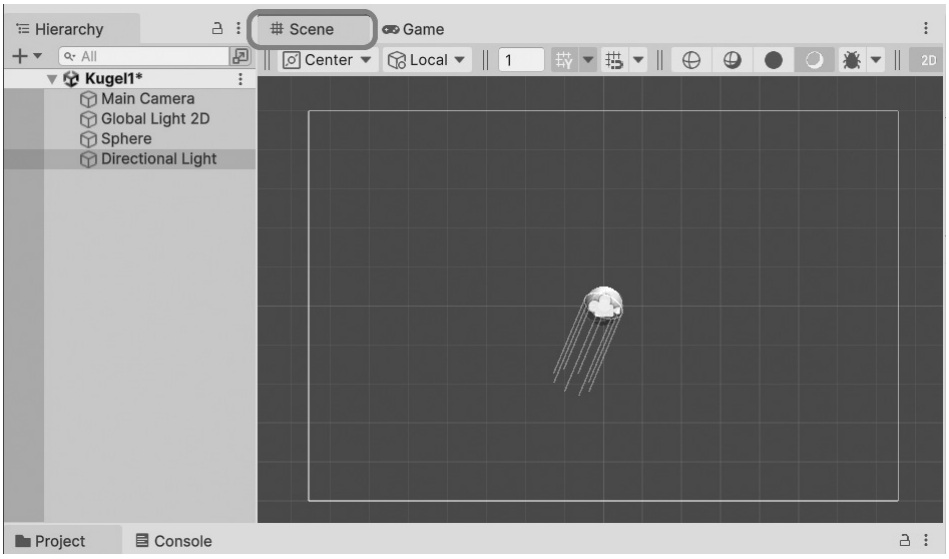


Kurz darauf sehen Sie die Kugel in einem anderen Licht. Erkennt man das nur im Game-Fenster?



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.



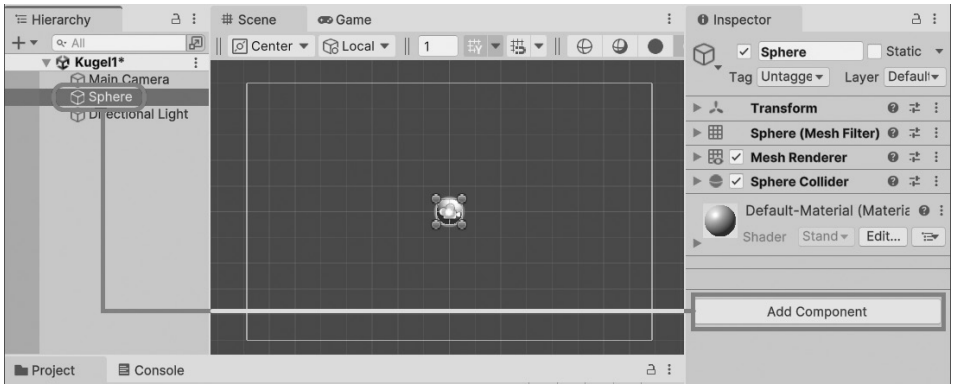
Nun gibt es ein weiteres Objekt, und es liegt ebenso wie die Kugel in der Fenstermitte.

Licht-Objekte

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

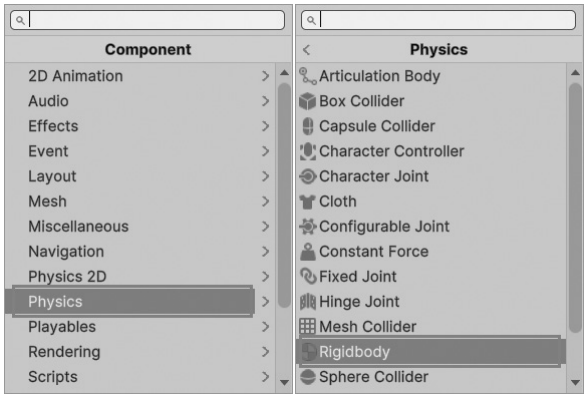
Toll wäre es, wenn sich die Kugel bewegen würde. Doch wie kriegen wir sie dazu? Zuerst einmal sollten wir diesem »Ding« physikalische Eigenschaften geben. Dass das Objekt aussieht wie eine Kugel, heißt noch nicht, dass es sich auch wie eine Kugel aus einem bestimmten Material verhält.

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.



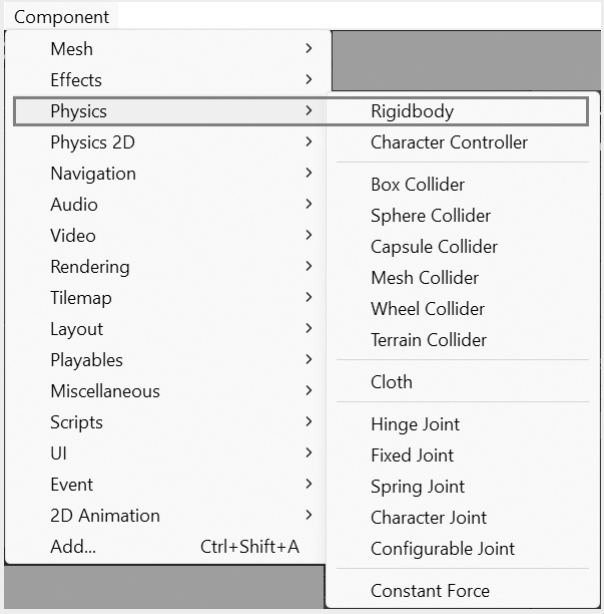
Ein kleines Kontextmenü öffnet sich.

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.

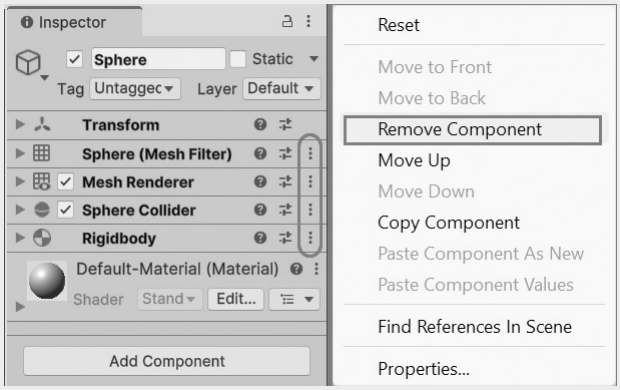


Damit bekommt die Kugel (Sphere) einige physikalische Eigenschaften wie *Masse* (unsere Kugel wiegt also auf einmal etwas) und *Gravitation* (sie wird vom Boden angezogen, würde also aus der Luft herunter auf den Boden fallen). Auch Kollisionen mit anderen Objekten und ihre Folgen sind nun möglich (ich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

Komponente wieder entfernen?

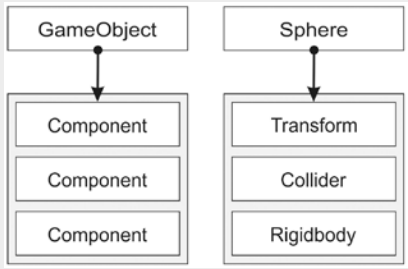
Falls Sie aus Versehen eine falsche Komponente hinzugefügt haben: Wie werden Sie diese wieder los? Schauen Sie im Inspector-Fenster mal genauer hin. Hinter jedem Komponenten-Namen sind ganz rechts drei kleine Pünktchen.

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.

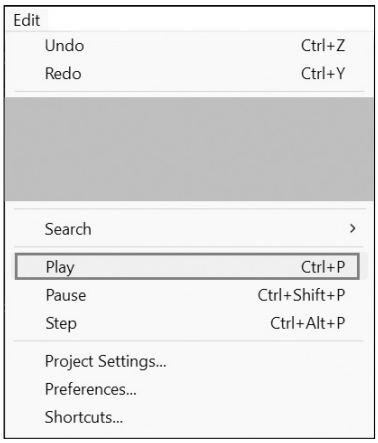


Die Komponente Transform hat jedes Spiel-Objekt »von Geburt an«. Weitere Komponenten lassen sich (fast) beliebig hinzufügen (aber auch wieder entfernen).

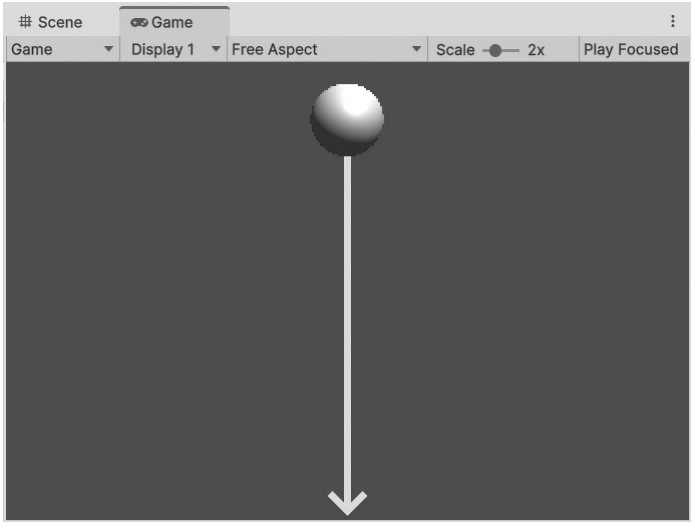
Wir sollten das Ganze schon einmal ausprobieren.

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.

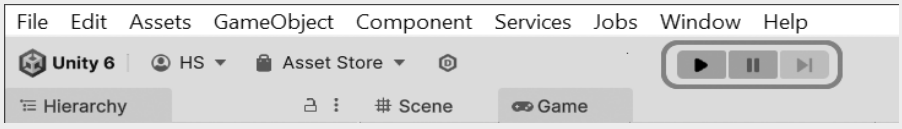


Im ersten Moment passiert anscheinend nichts, dann auf einmal fällt die Kugel und verschwindet aus dem Spielfeld.



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.

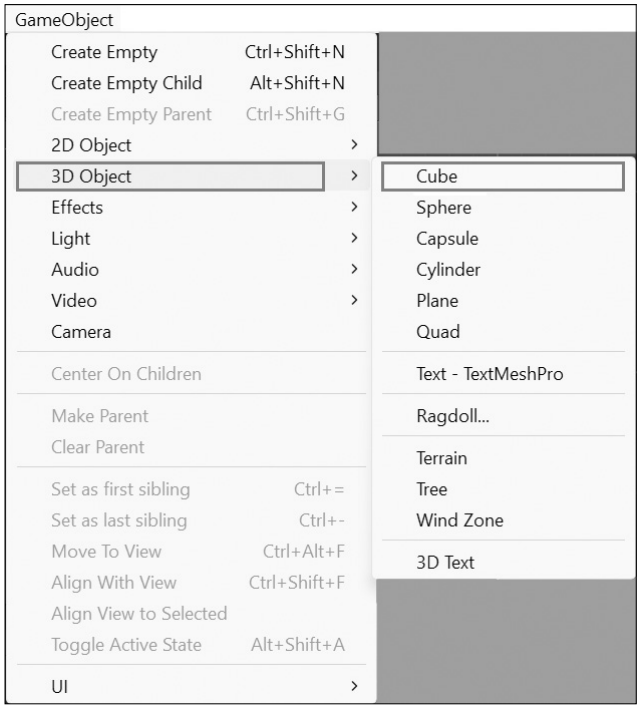
1.3 Gravitation und Kollision

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.

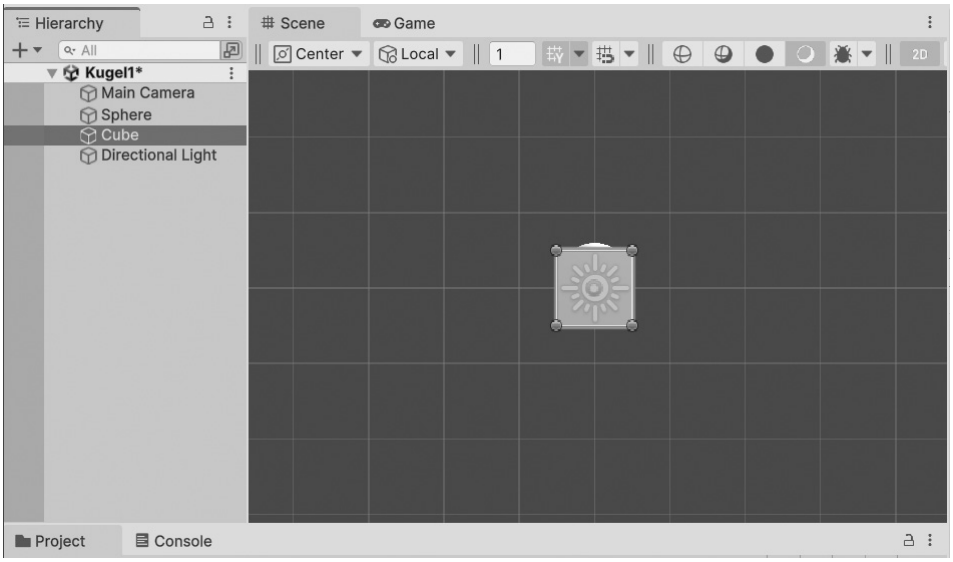
Andererseits kann die Gravitation einem beim Spielen nützlich sein. Denken Sie z.B. an ein Jump & Run-Game. Da geht es ja um Figuren, die springen und rennen, aber immer wieder irgendwo landen (und dazu brauchen sie die »Erdanziehungskraft«).

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.



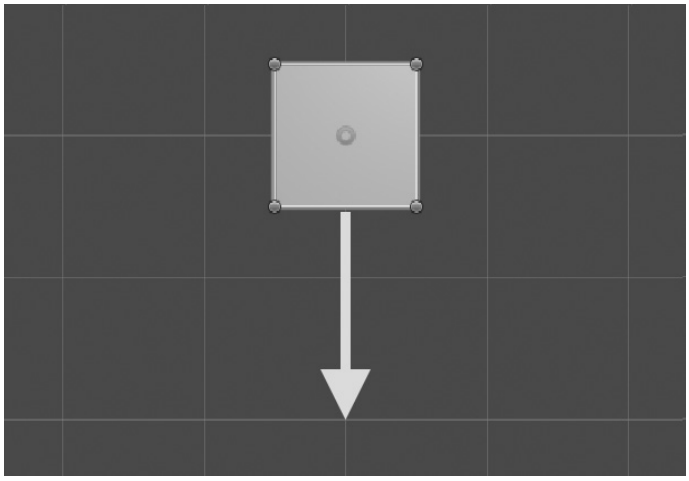
Im Scene-Fenster hat sich nun über den Kreis so etwas wie ein Quadrat gelegt. Auch hier zeigt der Inspector zahlreiche Informationen über das neue Spiel-Objekt.



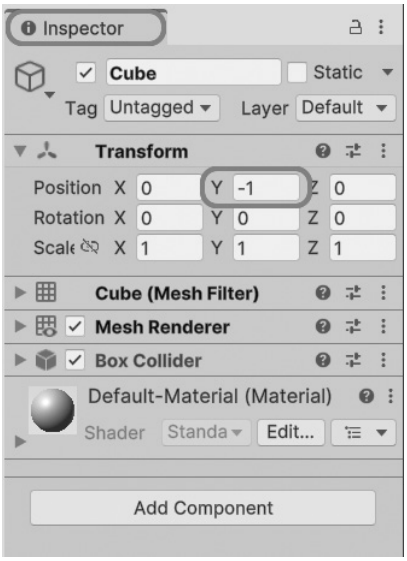
2. Vielleicht ist es sinnvoll, die Szene erst einmal wieder speichern (mit Klick auf File und Save).

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:

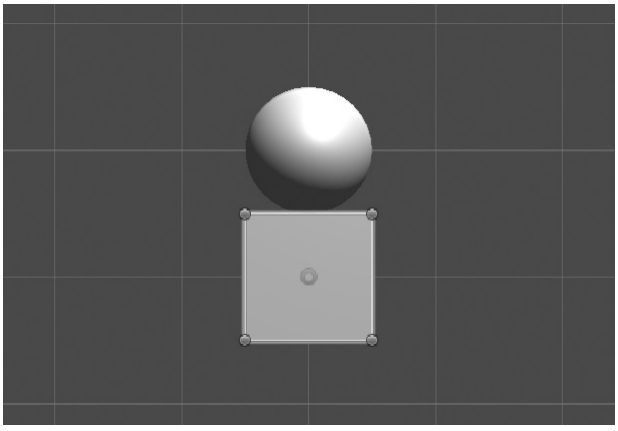
- Man packt das Objekt im Scene-Fenster mit der Maus und zieht es nach unten (das Ganze geht natürlich auch in andere Richtungen).



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



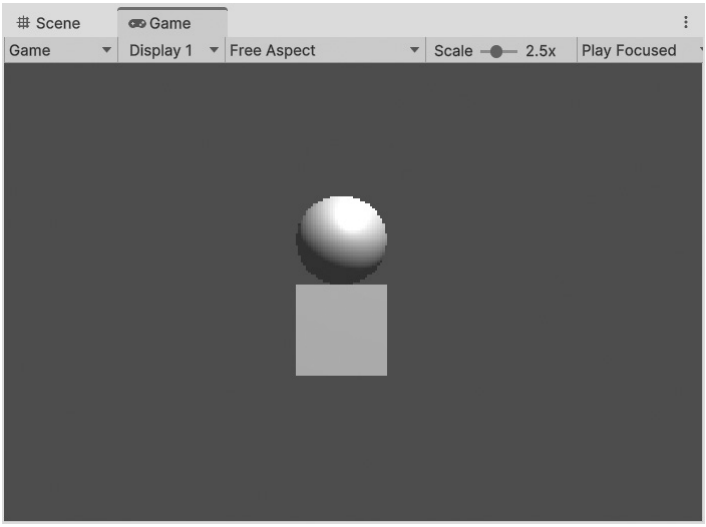
3. Sorgen Sie dafür, dass der Quader direkt unter der Kugel liegt.



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).

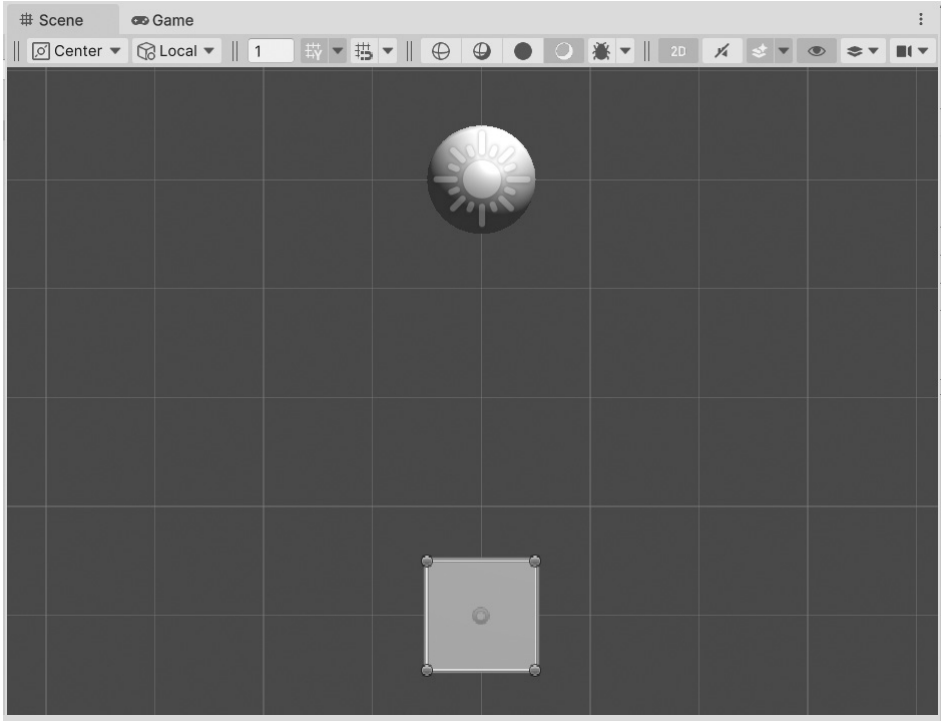
Zoom

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.



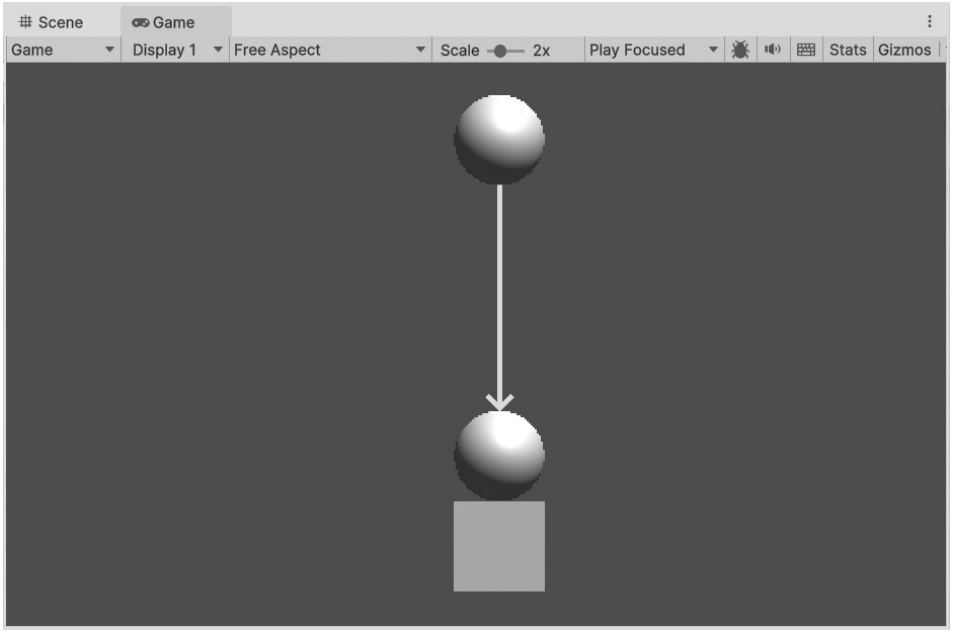
Bei mir steht im Inspector-Fenster hinter Position der y-Wert -5. Bei Ihnen kann da natürlich auch etwas anderes stehen.

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.

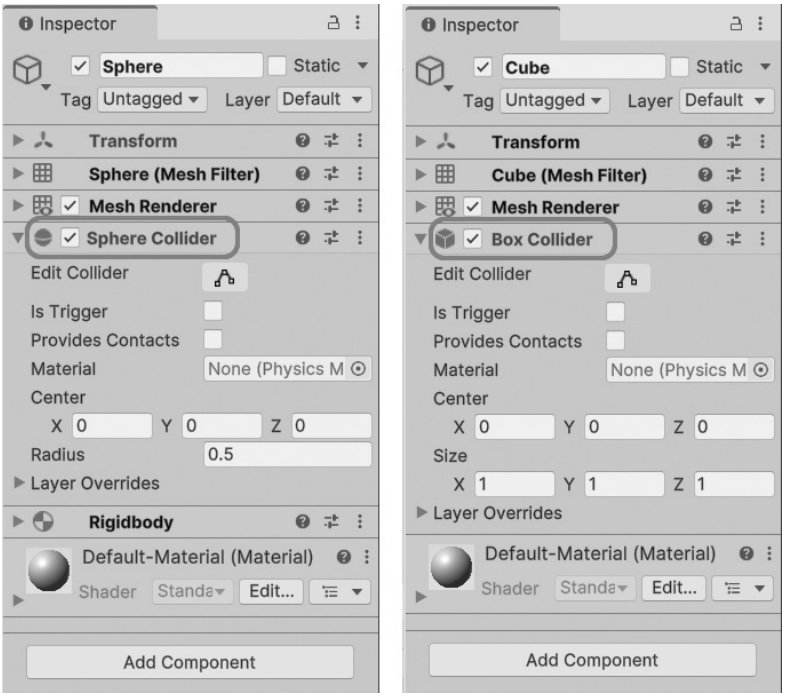
Wenn Sie wollen, können Sie die Kamera (Main Camera) mal anklicken, damit sie markiert ist. Dann lässt sie sich verschieben, ebenso wie eine Kugel oder ein Quader. Und damit ändert sich auch die Perspektive im Game-Fenster. Über Edit/Undo oder Strg+Z lässt sich diese Verschiebung wieder rückgängig machen.

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.

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?

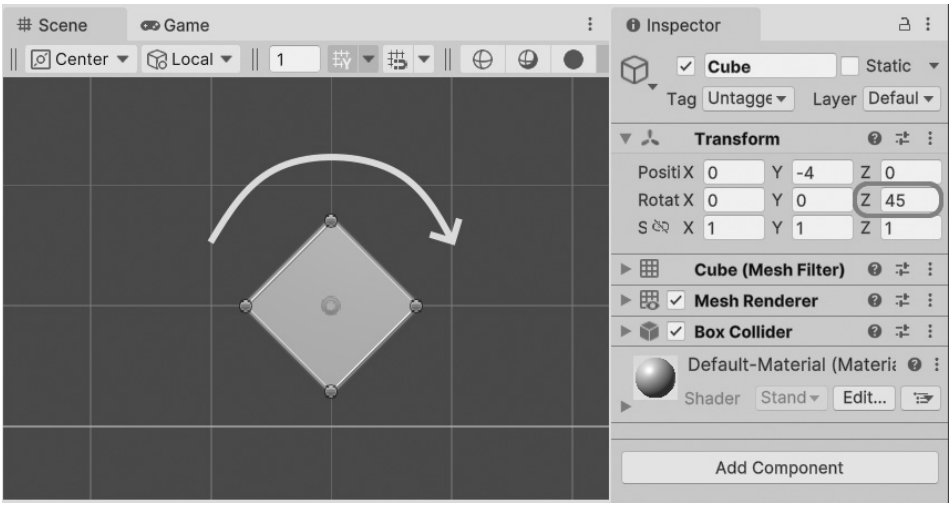
Halten wir jetzt erst einmal inne und schauen uns im Inspector-Fenster mal genauer um. Und zwar auf das, was unter Transform steht. Ich habe das in einer Tabelle zusammengefasst. So sieht es für ein Objekt aus, das gerade erzeugt wurde:

	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.

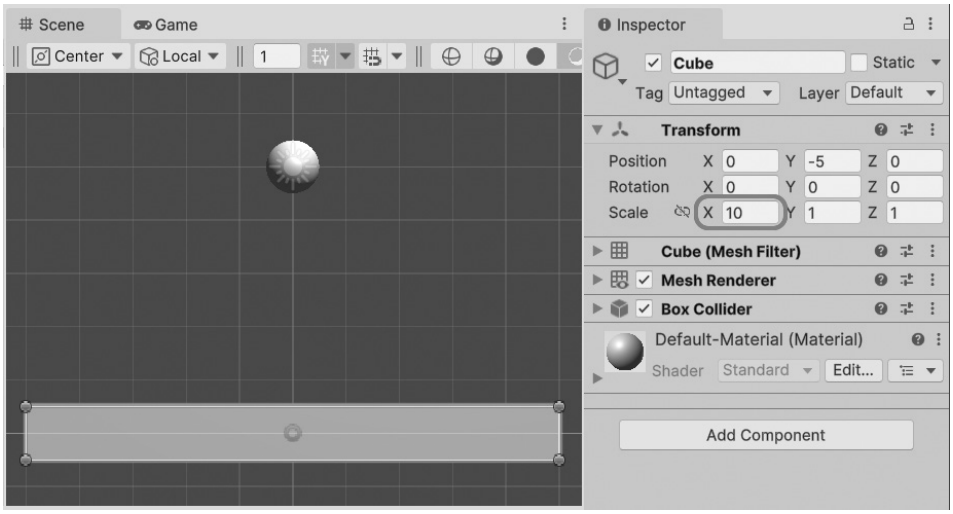


Und schon wird der Würfel um 45 Grad gedreht.

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

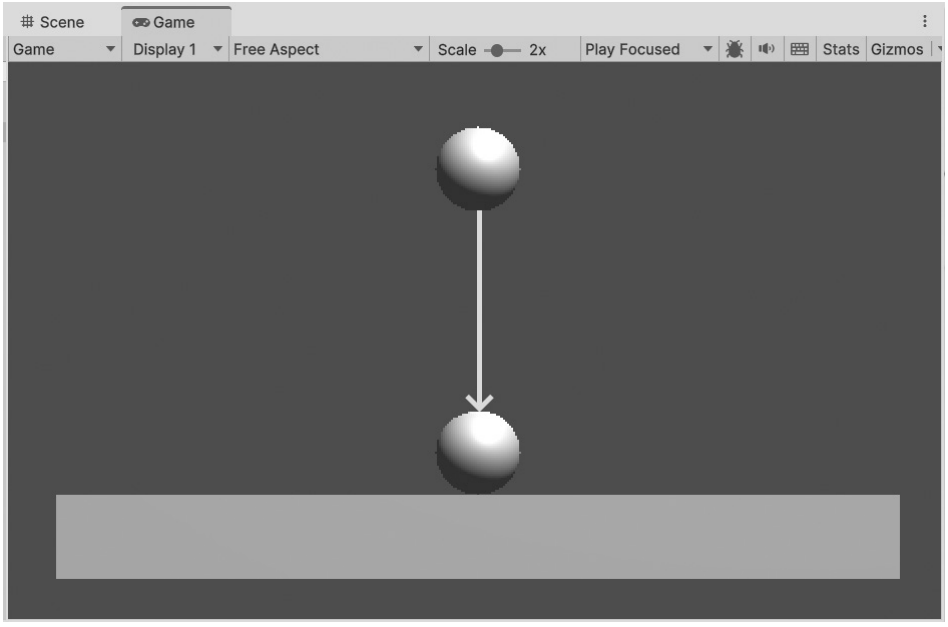
sieht ein bisschen mickrig aus. Warum machen wir aus ihm nicht einen Balken, der den ganzen unteren Spielfeldrand abdeckt?

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.

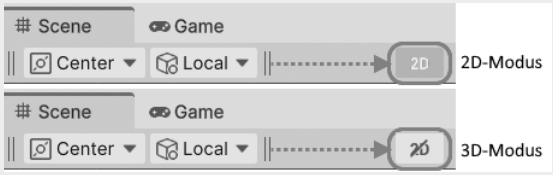
3. Speichern Sie nun die Szene, dann starten Sie das Spiel und schauen zu.



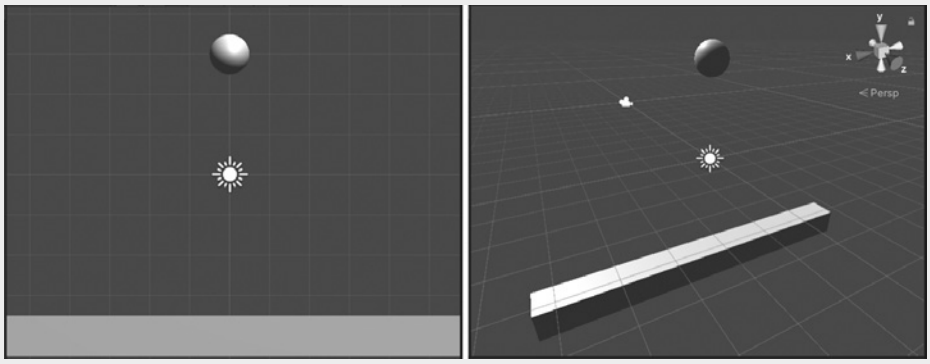
Ist das jetzt 2D oder 3D? Diese Frage haben Sie sich vielleicht schon viel früher gestellt. Genau genommen haben wir es die ganze Zeit mit Objekten zu tun, die dreidimensional sind. Da wir uns erst mal nur im 2D-Bereich (hier auf einer Fläche mit Höhe und Breite) bewegen wollen, haben wir beim Erzeugen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



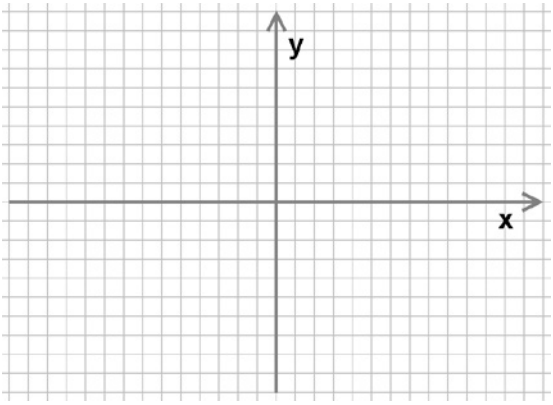
Hier lässt sich die Anzeige zwischen 2D und 3D umschalten. Im 3D-Modus ist die Anzeige 2D durchgestrichen.



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.)

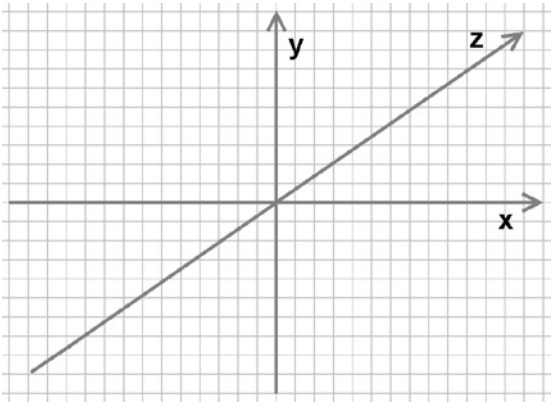
Das klassische Koordinatensystem besteht normalerweise aus der x-Achse (Horizontale bzw. Waagerechte) und der y-Achse (Vertikale bzw. Senkrechte).

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.

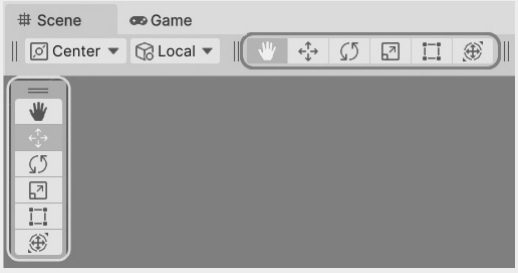
Wenn Sie Lust zum Experimentieren haben, dann schauen Sie der Kugel noch einmal beim Fallen zu, nachdem die »Bodenplatte« (bei gleicher Größe) nach vorn oder hinten verschoben wurde. Ergebnis: Die Kugel fällt weiter, weil sie ja nicht mehr auf den Quader trifft.

	X	Y	Z	Ergebnis
Position	0	0	1	Hinter der Kugel
Rotation	0	0	0	
Scale	10	1	10	So lang wie breit

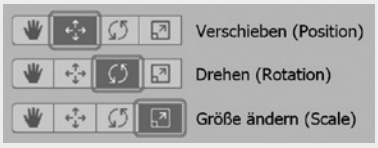
Sobald aber der Quader nach hinten und vorn ausreichend zu einer großen Plattform vergrößert wird, bekommt die Kugel wieder »Boden unter den Füßen«.

Transformationen mit der Maus

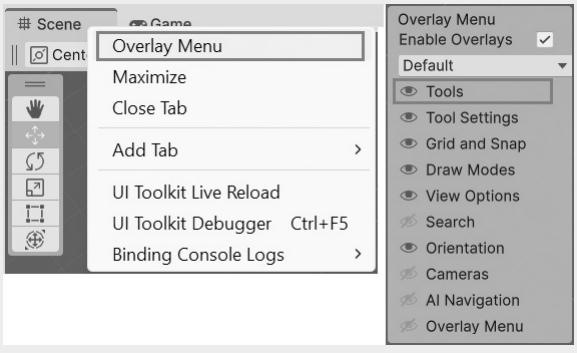
Wie man ein Objekt mit der Maus verschiebt, wissen Sie ja im Prinzip. Allerdings ist dazu eine bestimmte Einstellung nötig. Sie finden links oben im Scene-Fenster oder direkt unter dem Hauptmenü eine Reihe von Buttons.



Darüber kann man Objekte mit der Maus verschieben (Position-Modus), die Größe ändern (Scale-Modus) oder das Objekt drehen (Rotation-Modus).



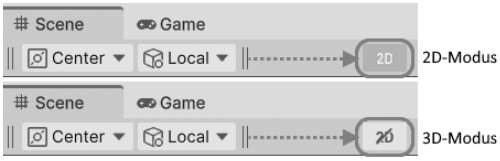
Die Umschaltung geht auch mit den Tasten W = Verschieben, E = Drehen, R = Skalieren.



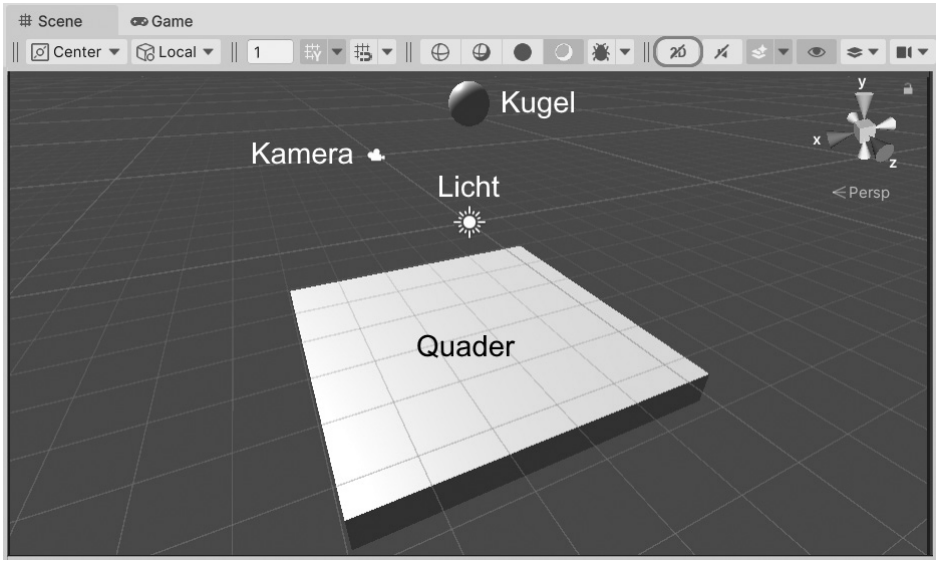
Falls Sie diese Buttons mal nicht finden oder sehen, lassen sie sich so wiederherstellen: Klick mit der rechten Maustaste auf den Scene-Reiter, im Menü den Eintrag Overlay-Menu und dann im nächsten Menü auf Tools.

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.

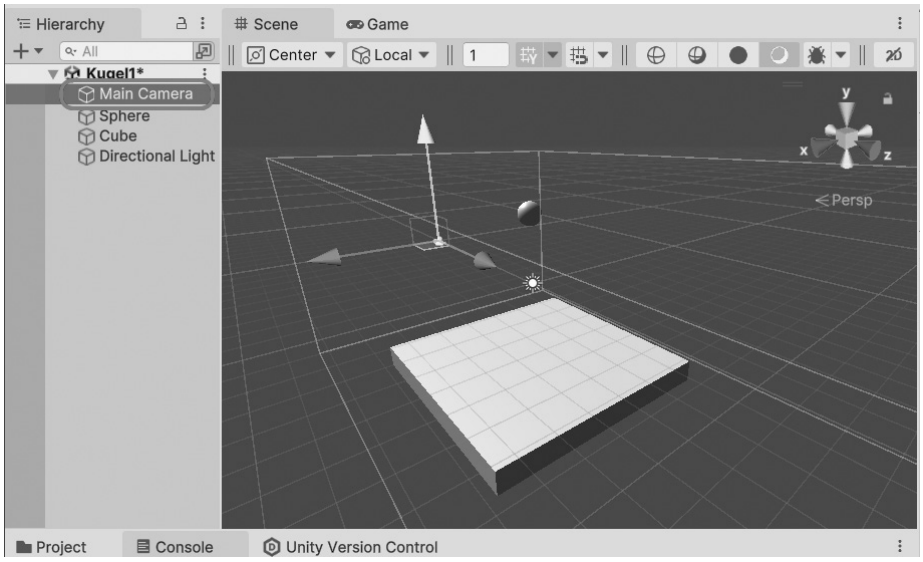


Die 3D-Ansicht der Szene hatten wir ja weiter oben schon mal. Man sieht die Kugel nicht in der Mitte liegen. Die Kamera schaut aus einiger Entfernung von weiter hinten zu. Beim Quader, den ich »Bodenplatte« genannt habe, sieht man die Ausdehnung in die Tiefe. Und ganz oben rechts wird die aktuelle Perspektive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



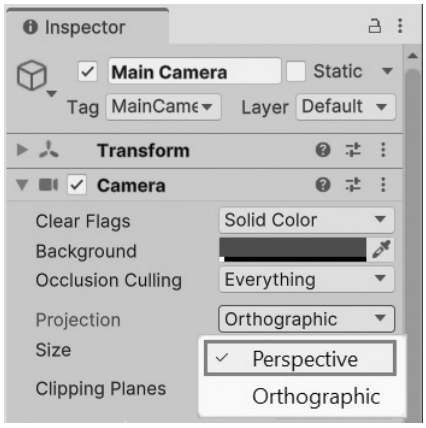
Um das Ganze auch mal im Game-Fenster zu sehen, müssen wir die Kamerasicht im Inspector-Fenster umschalten.

2. Markieren Sie dazu den Eintrag Main Camera links im Hierarchy-Fenster.

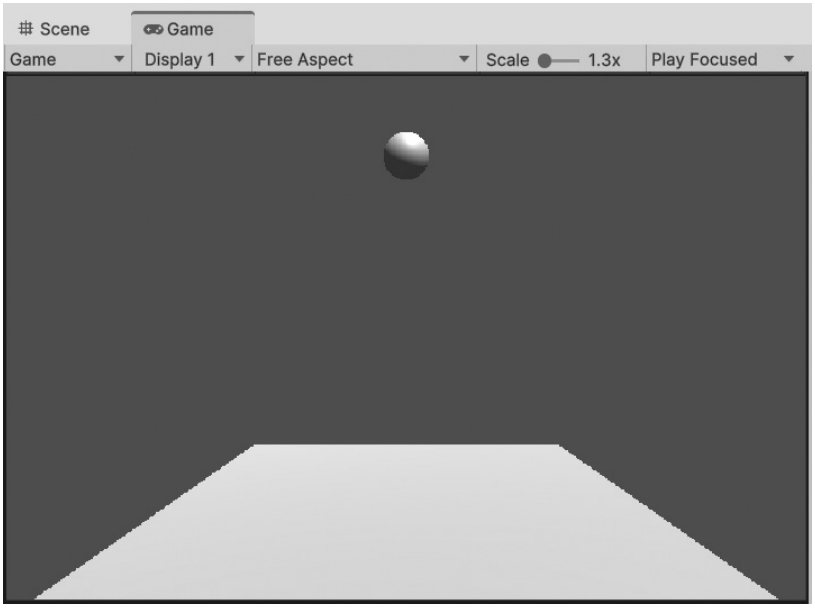


Würde ich jetzt ins Game-Fenster umschalten, wäre dort aber noch immer die 2D-Ansicht zu sehen.

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.

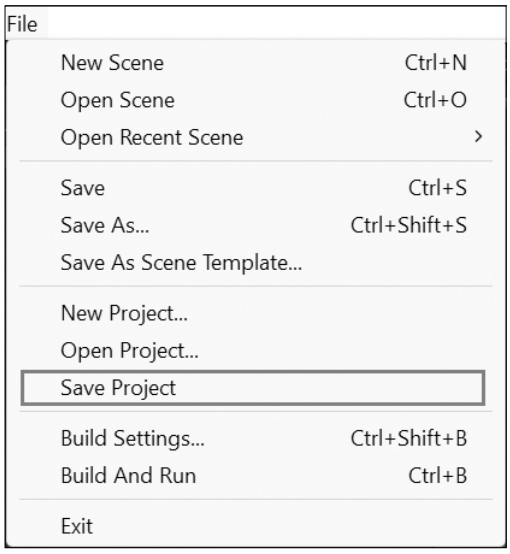
1.5 Ausblick

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.

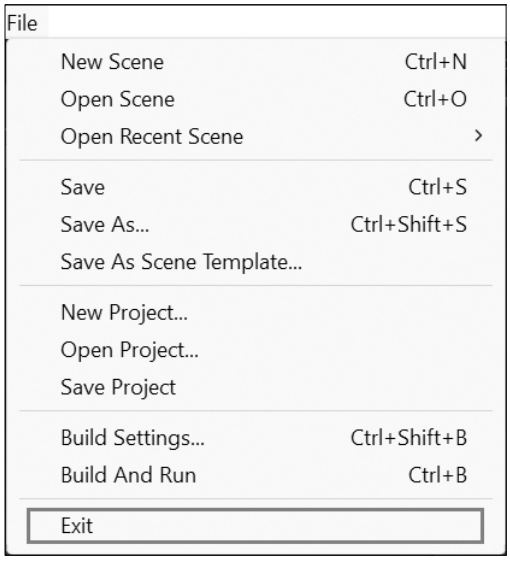
Nun kann man durch direktes Neueintippen den alten Namen überschreiben. Bei mir heißt die Kugel nun »Kugel« und der Quader bekommt den Namen »Boden«.

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:

GameObject	Spiel-Objekt, wie z.B. Quader oder Kugel, aber auch Kamera oder Licht
Collider	Komponente für Kollisionen
RigidBody	Komponente für physikalische Eigenschaften, wie z.B. Masse oder Gravitation

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

Transform	Komponente für Änderungen der Lage und Größe
Position	Verschieben in alle Richtungen (3D)
Rotation	Drehen in alle Richtungen (3D)
Scale	Vergrößern/Verkleinern in alle Richtungen (3D)

Stichwortverzeichnis

T

U

V