Unity ist eine Spiele-Engine des Unternehmens Unity Technologies, welche erstmals am 8. Juni 2005 veröffentlicht wurde. Es wird hauptsächlich für die Spieleentwicklung verwendet, allerding gibt es auch zahlreiche andere Anwendungsmöglichkeiten. Unity gibt Entwicklern, die Möglichkeit 2D, 3D Spiele zu entwickeln, aber auch AR/VR Anwendungen zu entwickeln. Die Primärsprache für das erstellen der Skripte ist C#, in der Vergangenheit wurde „Boo“ verwendet, welche eine stark anlehnende Syntax an Python hat. Unity ist die am weitesten verbreite Spiele-Engine mit einem Marktanteil von 45% stand 2017. 60 Prozent aller AR/VR Anwendungen wurden stand 2018 mit Unity entwickelt. Dies war auch einer der Gründe, warum wir uns für Unity entschieden haben.

https://digital.hbs.edu/platform-digit/submission/unity-engine-a-unicorn-powering-the-video-game-and-vr-ar-economy/

<https://de.wikipedia.org/wiki/Boo_(Programmiersprache)>

<https://de.wikipedia.org/wiki/Unity_(Spiel-Engine)>

**Unterstützte Zielplattformen**

Einer der Stärken von Unity ist die hohe Anzahl und Aktualität an unterstützten Zielplattformen. Es werden folgende Plattformen unterstützt:

PC-Betriebssysteme:

Windows, MacOS und Linux

Spielkonsolen:

Nintendo Switch, Wii U, PlayStation 5, Xbox One, …,

Mobile-Betriebssysteme:

iOS, Android, Windows Phone 8, …,

Webbrowser:

WebGL, Firefox, Safari, Chrome, Internet Explorer, …,

Anzumerken ist das die Spiele-Engine auch abseits der Spieleentwicklung Verwendung findet, zum Beispiel für Applikationen, welche von einer leistungsfähigen 3D-Grafik profitieren, wie jegliche Arten von Simulationen.

**GameObjects**

Das wichtigste Konzept im Unity Editor ist das Game-Objekt. Wenn man in der Szene ein Objekt hat, ist es ein „GameObject“. Mit speziellen Eigenschaften kann man es zu einem Charakter, einer Umgebung oder einem speziellen Effekt umwandeln kann.

Vordefinierte Komponenten geben dem Objekt Eigenschaften, welche dann als Licht, Gegenstand oder Kamera fungieren. Das GameObject kann man sich als einen leeren Topf vorstellen und die Komponenten als Zutaten. Mit der richtigen Kombination/Zutaten kann man das richtige GameObject/Gericht kreieren.

**Prefabs**

Unitys Prefab-System stellt die Funktion zur Verfügung, um Game-Objekte zu speichern und wieder verwenden. In den Prefabs werden die Komponenten, Eigenschaften und angehängte GameObjects gesichert. Dieses Prefab dient dann als Template um neue idente Instanzen zu erzeugen. In den selbst erstellten Ordner „Prefabs“ (siehe Abb 1: Prefabs) werden alle erstellten Prefabs abgespeichert. Änderungen an einen Prefab werden nach denn speichern von allen Instanzen übernommen (siehe Abb 2: Änderungen Prefab). Prefabs müssen allerdings nicht immer ident sein, es gibt durchaus die Möglichkeit Einstellungen individuell zu überschreiben, um zum Beispiel schadens- oder lebenswerte zu speichern. Um ein Prefab zu erstellen das gewünschte GameObject aus de Szene in den gewünschten Ordner ziehen.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abb. 1: Prefabs

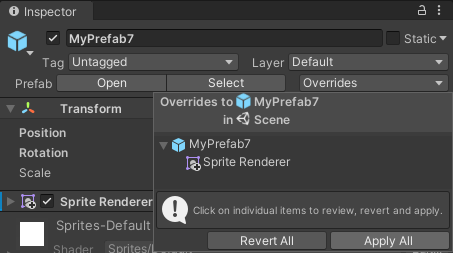


Abb. 2: Änderung Prefab

**Unity Kamera**

Kameras haben den Nutzen, dem Endbenutzer ein Bild bereitzustellen. Eine statische Kamera stellt einen bestimmten Teil der Szene aus einem festgelegten Blickwinkel dar, diese Option haben auch wir verwendet. Eine weitere Option ist die Kamera aus der Sicht einer benutzergesteuerten Figur zu platzieren, dies wird auch „first-person“ Perspektive genannt. Es gibt noch weitere Attribute zur Konfiguration, eines der Wichtigsten ist das Attribut Clear Flags, welches das verhalten der Kamera festlegt wenn sie ins Leere blickt. Die Standardeinstellung Skybox, legt einen blaue Himmellandschaft in den Hintergrund. Die zweite praxisrelevante Option ist „Solid Color“, hier wird eine Fläche mit ausgewählter Farbe gefüllt.

<https://docs.unity3d.com/Manual/class-Camera.html>

**Sprites**

Sprites sind 2D Graphik Objekte, welche durch spezielle Modifikationen und Techniken stark einen 3D Objekt ähneln können, falls man daran gewöhnt ist mit 3D Objekten zu arbeiten. Unity bietet mehrere Werkzeuge zur Verfügung, um mit Sprites zu arbeiten.

**Sprite Creator:**

Der Sprite Creator bietet die Möglichkeit Platzhalter Sprites zu erstellen, um an seinen Projekt weiterzuarbeiten wenn man seine gewünschte Grafik noch nicht besitzt.

**Sprite Editor:**

Mit dem Sprite Editor kann man aus einem großen Sprit mehrere kleine Sprite-Grafiken extrahieren. Das Ergebnis ist eine Reihe von Komponentbildern (siehe Abb. 3: Sprite extrahieren). Ein Anwendungsbeilspiel ist, wenn man eine Grafik hat mit allen Bewegungsvorgänge einer Figur und diese sollten separat gespeichert werden. Der Editor bietet die Möglichkeit eine Grafik in gleich große Zellen oder Pixel zu zuschneiden. Um diese Funktion nützen zu können muss der „Sprite Mode“ auf „Multiple“ gestellt werden (siehe Abb. 4: Sprite Inspector)

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abb. 3 Sprite extrahieren

**Sprite Renderer:**

Sprites werden mit einem Sprite Renderer und nicht mit einen Mesh Renderer gerendert. Dieser dient für die Darstellung von 2D- sowie 3D-Sprites in der Szene.

**Sprite Packer:**

Der Sprite Packer optimiert die Leistung des Videospeichers für das Projekt.

**Sprite importieren**

Sprites sind im Unity Projekt ein Asset. Über die Projekt anzeige kann man sie ansehen und verwenden. Es gibt zwei Möglichkeiten Sprites zu importieren.

1. Die Grafik aus dem Explorer (Windows) oder Finder (Mac OS X) in den Asset-Ordner kopieren. Unity erkennt diese Grafik automatisch und zeigt sie im Inspektor an.
2. In Unity unter Assets -> Import New Asset kann die gewünschte Grafik ausgewählt werden, diese wird dann im Inspektor angezeigt.

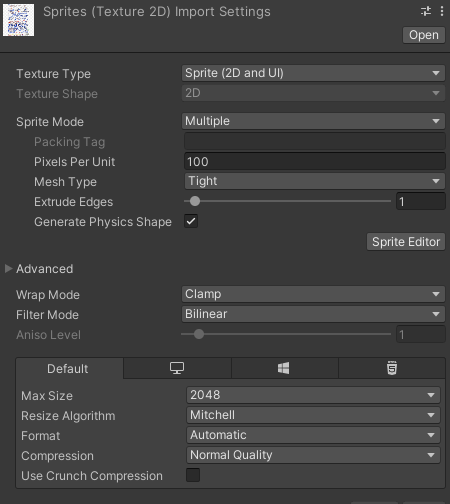


Abb. 4: Sprite Inspecotr

<https://docs.unity3d.com/Manual/Sprites.html#:~:text=Sprites%20are%20a%20type%20of,in%20Glossary%20in%20Unity%20projects>.