Creating a 3D game with Unity3D

Boris Fuchs, David Mitterlehner, Anika Seibezeder

2. Juli 2018

Contents

1 Prerequisite

1.1 Downloading Unity3D

Before we can make our game, we have to download the Unity 3D engine and editor. We can do this by going to the offical site from unity [4] and choosing a version to download. The versions to choose from are *Personal*, *Plus* and *Pro*. *Personal* is for beginners, *Plus* is for serious creators and *Pro* is for professionals. It is recommended to download the *Personal* version, as it is free to use.

While installing unity, you may be asked, which Platforms and Standard Assets you want to use. Figure 1 shows exactly what we need for our project.

After finishing the installation, you have to register an account and log in to use the editor.

1.2 Preparing Unity3D

For this tutorial we will build a simple environment, for which it is required to download the basic asset-pack of Unity3D first, and then add some premade characters. To add the premade objects to our project, we have to open
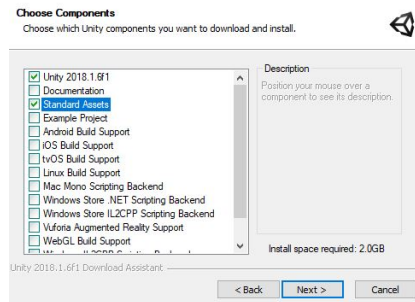
Figure 1: Unity Components

the Unity Asset Store in the editor by clicking *Window → Asset Store*. You may be asked to log in first, before using the store. The basic asset pack can be found under *Unity Essentials → Asset Packs → Standard Assets* [7]. The characters used in this tutorial can be found under *3D → Characters → Humanoids*. We will use the prefab called *Strong Knight* [5] as our knight character and *Zombie* [6] as the enemies chasing the character. After clicking *Add to My Assets* and finishing the download, the assets can be used in the current project.

2 Creating an Environment

The following section focuses at creating a simple environment with Unity3D. Unity3D provides an asset store, where all necessary assets (environment, scripts, characters,...) can be obtained. Anyone can offer his asset-packs in the store and other users can buy this asset-pack then. The prices of these asset-packs have a high variation, the range is from free usage to very expensive.

2.1 First step

Now its time to build a first terrain, the foundation of the prospective 3D-Models. A scale of 150 to 150 is an adequate base to represent the specified environment. As on fig.1 shown, the user can modify the scale at the last button in the terrain info template. All necessary (for this project) modifications can be done at the terrain info template. Take a look at the tabs that are interesting for crating a small environment. The first index tab is responsible for raise/lower the terrain. (e.g. for building mountains) The next two tabs can be ignored for this task. The fourth tab is for painting textures, and with the fifth tab (paint trees) the user is able to place trees at

the terrain. The sixth tab is for painting details, more precisely for "painting" grass. With the last tab its possible to change the settings of the terrain, as before mentioned, to modify the scale. The next steps take a closer look at the various index tabs.
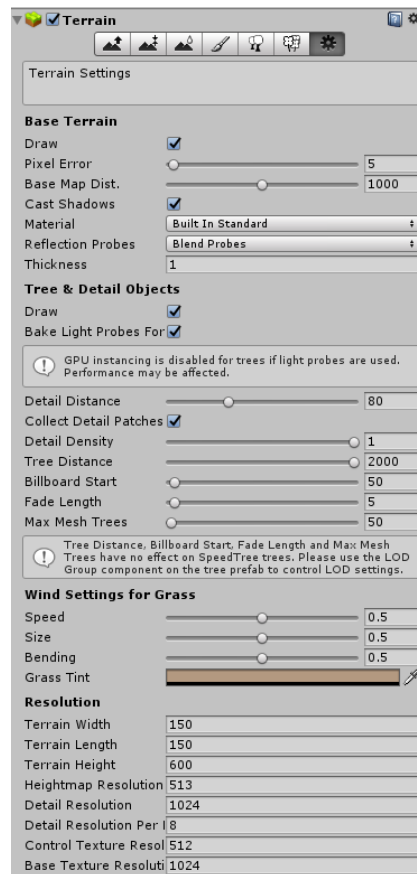

Figure 2: Terrain Info

2.2 Raise/Lower Terrain

This index tab has multiple brushes for lowering or raising the terrain. (see fig.2) The best way to find out which brush is the best for altering the terrain, trying all brushes out and checking the effects will help. Additional there are settings for the brush where the user can change the brush size and the brush opacity. Depending on what the user wants to handle, these settings can be fit on every situation. To raise the terrain only click with the left mouse button and hold until the terrain is high enough. For lowering the user has to press and hold shift before clicking the left mouse button.
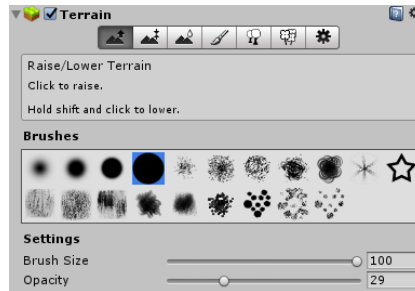
Figure 3: Raise/Lower Terrain

2.3 Paint Textures

One of the most important points is painting the textures. Without textures the most objects can not be identified. Like on the tab before here are also different brushes available for selection. The next step is creating a new texture. To choose between the different textures of the Unity3D standard asset pack, the user has to click on *edit textures → add textures → select.* If everything worked, a view shows all available textures. The next two steps add the texture to the current project, first double click for adding the choose texture, then press add. Now the user can repeat this instructions for adding as much textures as want.
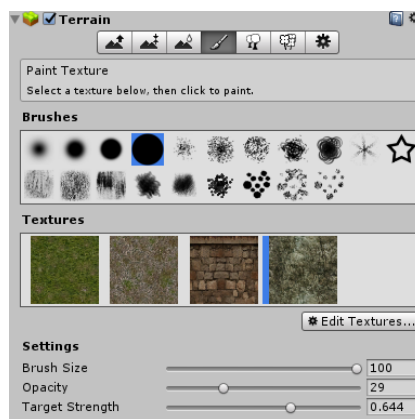

Figure 4: Paint Textures

2.4 Paint Trees

Inclose varying trees to project is nearly the same procedure as discussed in the chapter "Environment: Paint Textures". (*edit trees → add trees → tree prefab → choose tree → double click → add*) Its possible to paint only one

4

tree or to paint a lot of trees, depending on the brush size and the opacity. Now a lot of options are available. The key-features are: Adding other trees, mixing different trees and place it with a high brush size, place only a few trees or placing a dense forest depending on the current set opacity,....
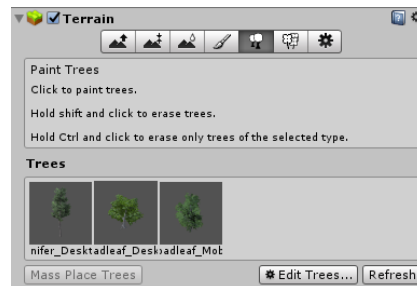

Figure 5: Paint Trees

2.5 Paint Details

The last section describes how to place grass on the terrain. The method to handle that is almost the same as before. Unity3D try's to stay abreast of the same to guarantee a easy handling for users. The user has to click once again on (*edit details → add grass texture → tree prefab*). After that, a small window appears where the user can change various settings. Inter alia it contains the minimum and maximum width and height plus settings to modify the color of the grass. (color can be change later, also when the grass is already placed) Selecting a grass is possible by clicking on detail texture and then double click on the decided grass texture like before.

2.6 Conclusion

The user is now able to deal with the basic features of Unity3D creating an environment. But Unity3D provides much more features, settings and possibilities to create beautiful environment combined with an easy and clear handling. This short description illuminated only the recommended points to create a simple terrain with grass, trees and textures.

3 Adding Character Controls

Character controls are a must have when developing a game. Otherwise the user would not be able to interact with a given character. No matter how
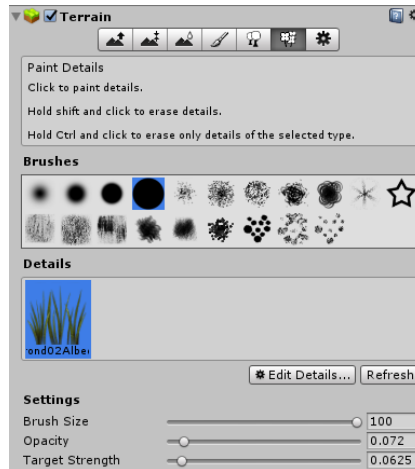
Figure 6: Paint Details

hard the user is pressing a key or moving the mouse, the character still would not move at all. With Unity3D the user interaction can be easily made by implementing scripts and adding them to the editor via Drag and Drop. The scripts can be written in C#, JavaScript and Boo [1]. In the editor itself there is no opportunity to write the code, but there are several supported programming editors such as the recommended editor MonoDevelop by Unity3D or VisualStudio by Microsoft [1].

For this tutorial we will use the knight character we downloaded before and combine him with the *Third Person User Control* Script from the Standard Assets.

3.1 Adding a Character

In Unity3D adding a character is easily done by dragging a Prefab or an object into the scene screen to the wanted position. There is also another possibility by dragging the Prefab or object into the hierarchy screen, where all used objects are shown. A Prefab is a pre-built object combined of one to many components [2], like the knight we downloaded before. To see and edit features of our knight, a left click on it on the hierarchy screen is needed.

6
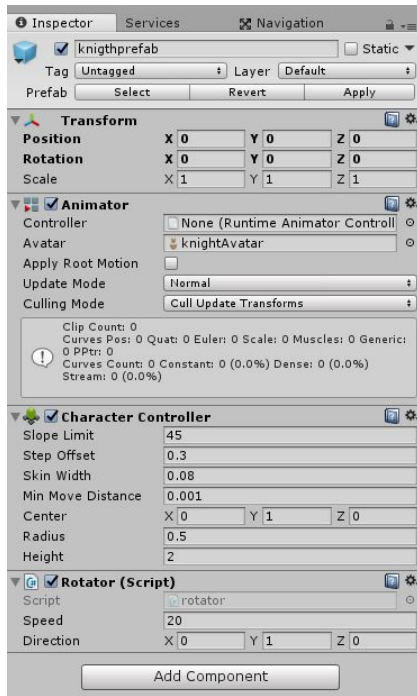
(a) Knights Components before
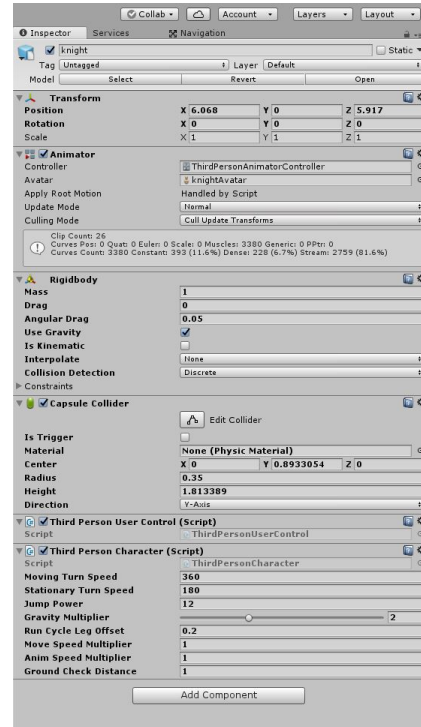
(b) Knights Components after

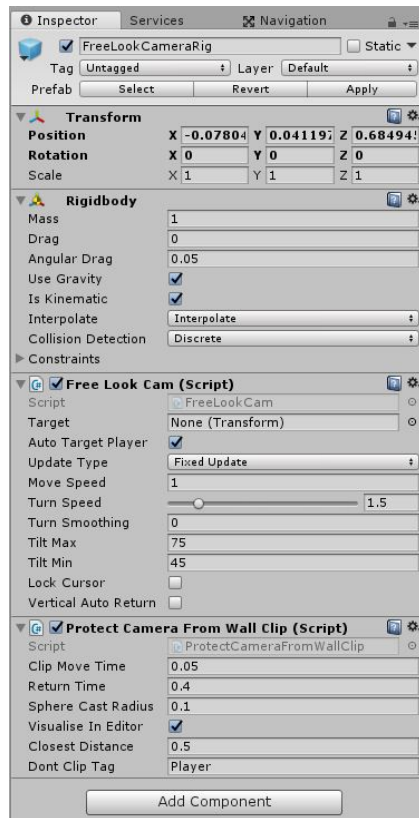Figure 7: A Comparison of what Components a character needs

When we first add our knight character, his components look like shown in figure 7a. Some of the given components of the knight, like the *Character Controller* and the *Rotator*, can be removed. The knight needs other components like a *Rigidbody*, which gives him a mass, a *Capsule Collider*, which triggers interrupts when colliding with other objects and the *Third Person Scripts*, which are premade by the Unity3D standard Asset Library. Figure 7b shows which components the knight should have.
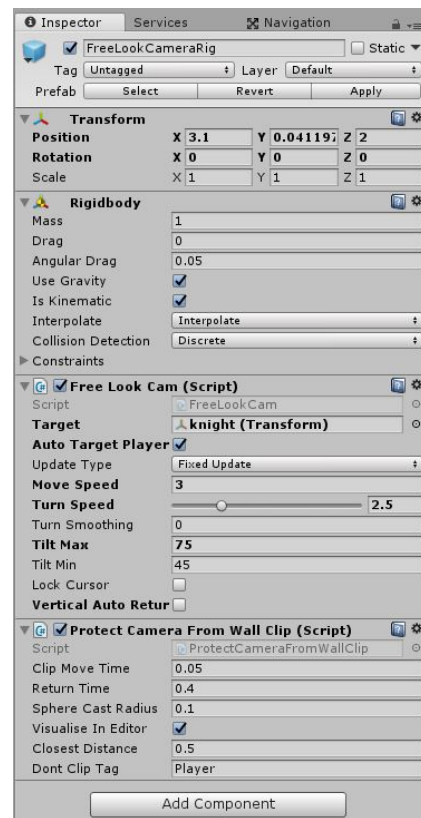
3.2   Changing the Camera

At the beginning of our example there will be a main camera in the hierarchy screen. Cameras in Unity3D are used to display specific content to the user. The main camera in Unity3D is a static camera which means, the camera doesn't move when the character is moving. But in many Roll Playing Games, the camera is following the character.
To implement this first delete the main camera and then go to *Assets →
Standard Assets → Cameras → Prefabs* and add the *FreeLookCameraRig* to

7

the scene by dragging it into the hierarchy screen. First thing to change is the position of the camera. We want the camera to be behind the character and we can do this by simply dragging the camera in the scene screen to the specific position. Next thing to do is to choose our knight character as the target at the *Free Look Cam* script. We also can change the move or turn speed of the camera to make the transitions faster.

(a) Camera Components before   (b) Camera Components after

Figure 8: A Comparison of what Components a camera needs

4   Adding Animations

The Unity editor supports various ways of adding animations to an object. The object can be simple, like a sphere, or complex, like a humanoid character with multiple joints.

4.1 Complex character animations

Complex animations, like character arm and leg movements, etc. can be obtained from the asset store. For example, there is a zombie asset on the store which includes walking, idle and attacking animations. An animator is also provided, which defines when, and under which conditions the animation states should be changed (Figure 9 and 10). To make the zombie actually walk towards the player and attack it, a script needs to be attached to it, for example the "AICharacterControl" script from the standard assets, where the player (knight) can be set as the target (See Figure 11).
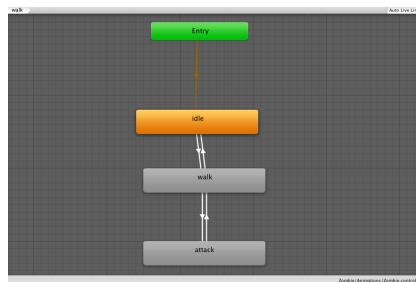
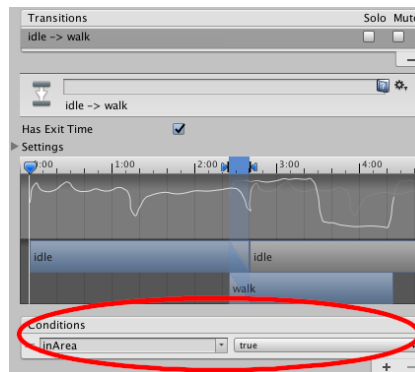

Figure 9: State diagram for the zombie character

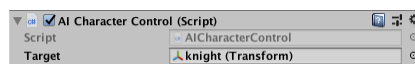Figure 10: Transitioning conditions for the zombie

Figure 11: Script for the zombie

9

References

[1] Ji Won Oak and Jae-Hwan Bae, "Development of Smart Multiplatform Game App using UNITY3D Engine for CPR Education", vol. 9, no. 7, pp. 263–268, 2014.

[2] Unity Technologies. "Manuel:Prefabs", Publication: 2018.1-002B, 2018.

[3] https://unity3d.com/unity

[4] https://assetstore.unity.com/packages/3d/characters/humanoids/strong-knight-83586

[5] https://assetstore.unity.com/packages/3d/characters/humanoids/zombie-30232

[6] https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-32351