GAME ENGINE COMPARISON

Kurs: S0012D, Spelmotorarkitektur

Morgan Nyman
mornym-9@student.ltu.se
2021-03-01

Introduction

In this report I will make a comparison between three game engines used to create games. I will start with a short introduction to each one of the game engines and then I will make a summarizing comparison.

Unreal engine 4

Unreal engine is the oldest of the tree game engines. It is developed by Epic games. It is one of the most used engines in the AAA space. Used in the development of advanced games such as Fortnite. Gears of War and Rocket League (Chris, 2021).

It has been in use for a long time, therefore it has a big community and you can get good advice from many users (Chris, 2021).

It's a powerful Engine, which have many useful features. The many features can be overwhelming for a beginner, but they can be very appreciated when you have been using it for a long time (Chris, 2021).

The coding language for Unreal is C++. Some people find it difficult to learn. It also includes something called blueprints, which is a visual scripting system that allows a person to create content without knowing how to code. However, some think it is infeasible over a longer time (Chris, 2021).

Unreal has an asset store where you can browse assets which other developers have shared. Without much work it is possible to put them into your own game. It is helpful for filling the game environment with content (Chris, 2021).

Unreal uses royalties. The consequence is that if you create a game with unreal and the revenue is more than 1 000 000 US dollar, you have to pay Epic Games 5% of that (Chris, 2021). If you choose not to distribute your product it's royalty-free (Epic Games 4, 2021).

Unreal engine is free to use for everyone (Chris, 2021).

Godot

Godot has been around only for a few years, which has gained much attention. It is open source. That means that there are no royalties that you must pay (Chris, 2021).

Godot is easier to learn for a beginner than for example Unreal. You can create simple things easy and quick. The language used for coding is particular to the engine. It is a language similar-like Python called GDScript. With the language GDScipt it is considered easier to create things in Godot than to create things with C++ in Unreal. The interface is simple (Chris, 2021).

Godot, however, is not considered to be a powerful engine compared to Unreal (Chris, 2021).

There is a small asset store in Godot, much smaller than the asset store in Unreal engine (Chris, 2021).

Nebula

Nebula is an open source and free-to-use C++ game engine. It's based on Nebula3, a game engine from Germany. 2011s Nebula3 was reworked by Fredrik Lindahl, Gustav Sterbrant and Johannes Hirche to become Nebula (github/gsept/nebula, 2021). The languages used to code are C++ and python in runtime. The managing of many entities is, perhaps, even better graphics than Unreal engine. It is not easy to use for beginners.

There is no asset store in Nebula.

The software is free-to-use when you include Nebula's version of the MIT-License when changing, distributing, coping, or using the Engine, The Engine's software (github 2/gscept/nebula, 2021).

Redistribution and user in source and binary forms are permitted if some conditions are met listed in the license text (github 2/gscept/nebula, 2021) .

Summarizing comparison

With the Nebula engine the programmer interacts with the engine only by scripting. With Unreal Engine 4 scripting is optional. With Godot both scripting and visual interaction with the engine is possible.

With Unreal Engine 4 it is possible to find and modify core components (source access). If you have an account and has stated your intention with the core

components (Epic Games, 2021). With Nebula it's possible to modify source components. With Godot it's allowed to modify the source components for personal or commercial use.

To add your own resources in all three engines you can place the resources you want to add in the projects root directory. In Nebula it is the only way to do it, whereas in Godot you can drop your assets directly in the project folder (Multipanikiller-studio, 2021). With Unreal Engine 4 it is possible to import using drag-and-drop, by using the import button or by right clicking (Epic Games 2, 2021).

Unreal Engine 4 have its own tool for building which is UnrealBuildTool (Epic Games 3, 2021). Godot uses Scons as building tool, which runs in python (Juan Linietsky, 2021). Nebula does not have its own building tools. Instead it uses Make, Ninja, CMake, VS-CMake, VS-Ninja, clion, and/or xcodebuild.

In summary, Unreal engine 4 is quite difficult to learn for a beginner, but it is a very powerful engine which can create more impressive 3D environments than Godot or Nebula.

Godot is more suited for beginners and amateur game developers. It is also more suited for small games (Chris, 2021).

Unreal engine 4 is more suited for AAA 3D games, whereas Godot is mor suited for 2D indie games (Chris, 2021).

Nebula is a relatively unknown game engine suited for experienced developers.

Referenser

Chris, G. d. (den 01 03 2021). *What Game Engine to use in 2020 (Unreal vs Unity vs Godot Comparison)*. Hämtat från youtube: https://www.youtube.com/watch?v=HDVR5l1pG4k

Epic Games 2. (den 01 03 2021). *Importing Content*. Hämtat från Unreal Engine: https://docs.unrealengine.com/en-US/WorkingWithContent/Importing/index.html

Epic Games. (den 27 02 2021). *What is GitHub*. Hämtat från Unreal Engine: https://www.unrealengine.com/en-US/ue4-on-github

Epic Games 3. (den 01 03 2021). *UnrealBuildTool*. Hämtat från Unreal Engine: https://docs.unrealengine.com/en-US/ProductionPipelines/BuildTools/UnrealBuildTool/index.html

Epic Games 4. (den 01 03 2021). *Unreal Engine*. Hämtat från FAQ: https://www.unrealengine.com/en-US/faq

github 2/gscept/nebula. (den 01 03 2021). Hämtat från https://github.com/gscept/nebula/blob/master/documentation/LICENSE

github/gsept/nebula. (den 01 03 2021). Hämtat från https://github.com/gscept/nebula

Juan Linietsky, A. M. (den 27 02 2021). *Introduction to the buildsystem*. Hämtat från godotengine: https://docs.godotengine.org/en/stable/development/compiling/introduction_to_the_build system.html

Multipanikiller-studio. (den 25 02 2021). Hämtat från Imort to Godot: https://www.multipainkiller-studio.one/dev-tips-for-game-developers/how-to-import-assets-from-blender-3d-to-godot-game-engine/