Nuake

Antoine Pilote

Oct 01, 2023

GENERAL

Welcome to the offical Nuake documentation website! Here you will find the most up-to-date documentation regarding the Nuake game engine. I strongly advise everyone to read up the general section to learn more about the project before cloning it.

CHAPTER

ONE

INTRODUCTION

Note: Nuake is a work-in-progress project and is not suited for profesionnal game development. Use at your own risk.

Welcome to the offical documentation of Nuake, an open-source 3D game engine! Nuake is a 3D game engine aimed to create games similar to quake with modern technologies. The recent resurgence of quake-like games gave me the initial ideas to create an engine designed to make those types of games. This means that integration of tools like trenchbroom and quake-based tools will be a priority during the development of Nuake.

1.1 What is Nuake?

Nuake is a work-in-progress game engine aimed to give a simple but flexible tool to create quake-type games.

1.2 What type of game is it for?

You can create any type of game using Nuake, but it is mostly designed to create 3D games. You should be able to create FPS or TPS games without problems, although it was originally designed to create first person games.

CHAPTER

TWO

BUILDING

2.1 How to build

Building Nuake is currently the only supported way to get Nuake. This means that you will need to build the projects file, build, then launch the project.

The following step should help you get Nuake running on your computer:

1. Clone the git repo and the submodules recursivly. This can be done using the following command:

   git clone –recurse-submodules https://github.com/antopilo/Nuake.git

2. Launch the cmd file called *generate.bat*. This should generate a visual studio solution for you.
3. Build the project, everything should build in the right order and launch the editor.

CHAPTER

THREE

CONTRIBUTION

3.1 How do I contribute?

You can simply fork the current github repository and create pull requests, I will look over them personally.

CHAPTER

FOUR

SCRIPTING

4.1 Entity scripts

Entity scripts are Wren scripts that are attached to an entity using the WrenScript component.

After attaching the WrenScript component on an entity, you can select the script file directly. In addition to pointing to the script file, you need to specify which class in the script is considered as the entity script.

The class chosen must inherit from ScriptableEntity.

Here is an example of a barebone entity scripts:

```
import "Scripts/ScriptableEntity" for ScriptableEntity

class TestScript is ScriptableEntity {
    construct new() {
    }

    init() {

    }

    update(ts) {



    }

    exit() {

    }
}
```

4.2 Interface scripts

4.3 Engine module API

4.4 Math module API

4.5 Scene module API

CHAPTER

FIVE

TUTORIALS

5.1 Project tutorial

5.2 Scene tutorial