Nephilim SDK

This game engine is written and maintained by Artur Moreira (artturmoreira@gmail.com).

Nephilim SDK

This software development kit can be considered a generic purpose game engine. It is aimed at programmers of all levels of expertise, as beginners can achieve great things easily as well as advanced users can leverage more complex features to making top quality games. You will be using C++ to program games with this library, even though scripting is possible natively with AngelScript.

One of the biggest aims of this tool is to provide portability. This means in essence a seamless way to program games for multiple platforms. The same source code will run just the same in platforms such as Windows, Linux, Mac, Android and IOS.

It makes possible to render hardware accelerated graphics in all these platforms, both 2D and 3D, using OpenGL and its mobile variants. Exactly to make the library versatile for a wide range of users, virtually no limits exist in the graphics system. You can use low level systems close to raw OpenGL as well as high level drawing facilities which do everything for you.

Also, all kinds of API's are provided along with the graphics. You will find ways to play sounds, to manage geometry, images, animations and other game specific tasks.

SFML

This library was initially made using SFML to its fullest but once the engine started to be ported to android and IOS, SFML wouldn't work well anymore as it doesn't support these platforms. Because of this, the use of SFML was slightly dropped as more progress was made in the library. At this point still some SFML's facilities are used in Nephilim, in some platforms, but the rest was re-implemented. Nevertheless, the library is heavily inspired on SFML and takes many ideas from it. This actually makes porting games made with SFML to Nephilim quite fast and easy, as the API's are very similar. Also, while some features were re-implemented, one can still find many fragments of SFML's source within the source code of Nephilim. For all these reasons together, SFML will always be a big part of this engine's development and features, even if it is not being used directly by it. So, this foreword works as a way of saying thanks to Laurent Gomila, the original SFML author, whose code helped this library come true greatly. I do not claim authorship over any source that may have been transposed from the SFML's source into Nephilim and this is a clear statement of Laurent's official authorship of any of those fragments.

Required tools

In order to take advantage of the SDK to its fullest, there are a few things you should have in your system before starting.

Premake

The premake build system is primarily used by the sdk for building itself and the samples. The build script targets the version *4.4beta4*, so it is important that you use that version or higher of premake in your system. This tool is amazing and will certainly make our life easier.

Git

The git repository management system is a great tool. GitHub hosts the sdk latest sources and allows many powerful features like pull requests and forking possible. While you don't absolutely need git in your machine to use the sdk, it will help a great deal when working with it.

Installing on Linux with premake4

This is a quick getting started tutorial for linux users. Here's a simple way to get Nephilim up and running.

1. Open a terminal and go to the desired directory
2. git clone https://github.com/DevilWithin/Nephilim.git
3. cd Nephilim
4. premake4 gmake
5. cd build/gmake/
6. make OR make config=release

And done! The library is built in /Nephilim/lib and the samples are built in /Nephilim/bin.

Note: If some sample is failing to compile you can just "make SampleName" to build a specific sample directly.

Installing on Windows with Visual Studio

Once you have the repository in your machine, you can:

1. premake4 vs2010
2. Open the project file in /build/vs2010/ and start building

Preparing an Android environment on Linux

Not optimized yet.

Preparing an Android environment on Windows

Not optimized yet.

Building in the Apple IOS system

Not optimized yet.