Welcome to the API Reference for the Lightning game engine.

**Contents  
1. General Overview**1.1. Lightning API style  
1.2. Namespaces  
1.3. Standard .NET types used  
1.4. The Renderable Class **2. Getting Started**2.1. Setting Up  
2.2. Creating a Window  
2.3. Window Settings  
2.4. Delta Time & Framerate Management **3. Modules**3.1. LightningGL  
3.2. LightningPackager **4. Scenes**4.1. The Scene Manager  
4.2. Creating Scenes  
4.3. Scene Management  
4.4. Scene Events  
4.5. Shutdown **5. Settings (Global & Local)**5.1. GlobalSettings Overview  
5.2. Global Settings Values **6. Cameras**6.1. Camera System  
6.2. Camera Types **7. Input & Collision**7.1. Input overview  
7.2. Key Input  
7.3. Mouse Input  
7.4. Hit Testing **8. Textures (and atlases)**8.1. Texture Loading  
8.2. Texture Drawing  
8.3. Texture API  
8.4. Texture Atlases  
8.5. Texture Blending **9. Animation**9.1. Animated Textures  
9.2. Animated Texture Blending **10. Lighting**10.1. Drawing Lights  
10.2. Screenspace Map  
10.3. Environmental Light **11. Particle Effects**11.1. Creating & Loading Particle Effects  
11.2. Playing Particle Effects  
11.3. Particle Effect Modes  
11.4. Particle Effect Parameters **12. UI**12.1. UI System  
12.2. UI Events  
12.3. UI Gadgets  
12.3.1. Buttons  
12.3.2. ListBoxes  
12.3.3: ListBoxItem  
12.3.4: TextBoxes  
12.3.5: CheckBoxes **13. Text Rendering & Font Management**13.1. The Font Manager  
13.2. Loading Fonts  
13.3. Drawing Text  
13.4. Text Properties **14. Localisation**14.1. Localisation   
14.2. Localisation Settings in Engine.ini  
14.3. Localised Strings **15. Audio**15.1. The Audio Manager  
15.2. Loading Audio  
15.3. Playing & Managing Audio **16. System Information & Feature Detection**16.1.The SystemInfo class  
16.2. Enforcing System Requirements  
16.3. Detecting Features **17. Packaging**17.1. Packaging Your Game  
17.2. The MakePackage tool  
17.3. Package File Options  
17.4. How to Run Your Game Packaged **18. Debugging**18.1. Debugging Your Game  
18.2. The FPS Meter  
18.3. The Performance Profiler **19. Advanced Usage (How to Interface with SDL)**19.1. Interfacing with SDL  
19.2. Lightning + NativeAOT  
19.3. Extending the Engine **20. Error Message Reference  
  
1. General Overview  
1.1. Lightning API Style**The Lightning API is based on the concept of windows, and, optionally, scenes.  
  
Each Window is a rendering context that allows you to use Managers.   
Managers are the various parts of Lightning that allow specific rendering operations, such as rendering lighting or input, to occur. It is generally required to pass the current Window being used to any rendering APIs that you call.

A scene is simply an “area” of a game and is an optional construct – the Scene Manager can be turned off with the **DontUseSceneManager** GlobalSetting. It provides ready-made basic handling of startup and shutdown as well as some very basic event handling.

**1.2. Namespaces**

**LightningGL**Provides all of the primary rendering APIs.  
**LightningPackager**Provides all of the packaging APIs.  
**NuCore.SDL2**Provides access to unmanaged SDL2, SDL2\_image, SDL2\_mixer, SDL2\_ttf, and my fork of SDL2\_gfx APIs and functions.  
**NuCore.Utilities**Provides utility APIs.

**1.3. Standard .NET types used**

**System.Drawing.Color**Defines an ARGB-formatted color.  
**Documentation**: <https://docs.microsoft.com/en-us/dotnet/api/system.drawing.color?view=net-6.0>

**System,Numerics.Vector2**Defines a two-dimensional vector.  
**Documentation:** <https://docs.microsoft.com/en-us/dotnet/api/system.numerics.vector2?view=net-6.0>

**1.4. The Renderable class**

Any renderable object is expected to inherit from the Renderable class and override its Draw method – this provides the object with a few basic properties, such as position and size, that can be manipulated in order to make drawing the object easier for the programmer. UI elements are extended by inheriting from the Gadget class.

Properties:

**SnapToScreen**Determines if the Renderable will be drawn in world-relative space or camera-relative space.