This is the release notes documentation for the Lightning game engine. This document’s goal is to assist you, the developer, in understanding all of the Lightning game engine’s latest functionality.

**Version 1.1.0**October 29, 2022

**General**

* Updated from SDL 2.0.22 to SDL 2.24.1.
* Updated to SDL2\_mixer 2.6.2 – MOD files are no longer supported due to bugs on SDL’s end, sorry, but there are many new functions.
* Updated to SDL2\_image 2.6.2 – JPEG-XL and AVIF image formats are now supported.
* Examples now use the installed engine instead of the version in the “shared” folder.
* All **UI events** can now be used by any renderable.
* The API has been standardised on American English in order to make it easier for developers who develop against other engines to adapt to Lightning. Please update your games accordingly.
* Unified the **MousePressedEvent**, **MouseReleasedEvent** and **MouseEvent** classes into one **MouseEvent** class.
* **NuCore.SDL2** has been renamed to **LightningBase**.
* Everywhere that previously referenced **frames** as a measure of time now measures **milliseconds**, such as the Particle Manager.
* AssetManager is now an actual base class for all asset managers (except LocalisationManager for technical reasons) and all asset managers can have **AddAsset** called on them with an Asset object to add and load an asset and **RemoveAsset** called on them with an Asset object to unload and remove it. The asset classes are called **\*\*\*AssetManager**, (such as TextAssetManager, etc) but for compatibility with Lightning 1.0 applications, they are instantiated under the 1.0 style property names. A global using is used to abstract this process from the developer in the project template.
* Fixed a bug in **Engine.ini** using the wrong value for the **ProfilePerformance** property.
* Fixed a bug where some globalsettings were parsed case-sensitively.

**Rendering**

* Refactored the renderer. It is now a lot more centralised and somewhat faster.
* Implemented **Z-Index** for renderables using the **Renderable::ZIndex** property. Renderables are only re-sorted when the order changes.
* Offscreen renderables are now culled. This increases performance.
* Fixed **Renderer::CurFPS** being an internal property.
* **Window** has been renamed to **Renderer**, and **WindowSettings** has been renamed to **RendererSettings**.
* Renamed the **Renderer** enum to **RenderingBackend**.

**Texture**

* Added **Texture::Access** so you can acquire a Texture’s access at runtime.
* Added **Texture::SetOpacity** to quickly set the Opacity of a texture.
* **TextureManager** is no longer experimental.
* Added **TextureManager::GetInstanceOfTexture** to clone a texture. The **clone** parameter will use the original texture’s handle if set to true. This has the beneficial effect of reducing memory usage, but will also result in **SetPixel** and **GetPixel** affecting every single draw of the texture.
* **AnimatedTextures** now obey delta time.
* Implemented **GetPixelGlobal** and **SetPixelGlobal** for AnimatedTextures.

**Text**

* Implemented font caching (with the internal **FontCache\*\*** classes). Text is now cached by the engine and automatically discarded when no longer used, resulting in a speedup of 20x in some cases. (Minesweeper went from 43fps to 550-800fps, and my Test Program went from 80-100 to 450-600fps!).
* **FontManager::DrawText** has been moved to a new **TextAssetManager** class, usable through **TextManager::DrawText**.

**Camera**

* Implemented camera shake using the **Camera::CameraShake** property.
* Implemented camera velocity using the **Camera::Velocity** property.
* Implemented the **Camera::AllowCameraMoveOnShake** property to disable the correction behaviour to keep the camera in the same place when shaking.
* Implemented the **Floor** camera type. This positions the origin of the camera at the bottom of the screen, as if a floor was rising.

**Packaging**

* Lightning now verifies the intended engine version of a WAD file instead of using a placeholder value, as in version 1.0.

**Animation**

* **A new animation engine has been implemented:**
* The new animation engine allows you to animate any property of any Renderable. There are currently 6 modifiable properties for animations, more will be added in future releases.  
  The animation format uses JSON files.
* Added the new Lightning Animation Editor (**AnimTool.exe**) that allows you to edit and generate animations.

**Settings**

* Moved both Global and Local Settings APIs to **LightningBase.**
* Implemented **LocalSettings::AddValue**, **LocalSettings::SetValue** and **LocalSettings::DeleteKey** to more easily manipulate LocalSettings.
* Local Settings is no longer an experimental API.
* If the GlobalSetting **DontSaveLocalSettingsOnShutdown** is **false**, Lightning will save local settings on shutdown.

**Lighting**

* Coloured lighting has been implemented using the **Light::LightColor** property.

**Particle Effects**

* Minor refactoring.
* Particle effects now obey delta time.
* Changed all **uint** properties of **ParticleEffect** to **int**.

**UI**

* Cursor blink now obeys delta time.
* Every Gadget property with “colour” is now “color” as part of the American English standardisation. This is to increase the familiarity of Lightning to game developers using Unity, Unreal, Godot, or similar game engines.

**Localisation**

* Added the GlobalSetting **LocalisationFolder** in order to redirect where localisation INIs can be loaded from.

**Version 1.0.4  
September 17, 2022**

* Change all remaining “LightningGL” branding to “Lightning Game Engine”
* Modify GlobalSettings values when changing position or size of window
* Fix minor typo in release notes documentation (“zThis” => “This”)

**Version 1.0.3  
September 4, 2022**

* Fix lighting screen-space map moving with the camera (catastrophic failure)
* Implemented **ParticleEffect::RemoveEffect**,which will unload and remove a particle effect.
* Fixed the Light Manager’s screen space map being unlocked twice each frame.

**Version 1.0.2  
September 3, 2022**

* Fix all UI elements being rendered twice. This will increase overall performance by 10-100% depending on how much UI is being used in your game.
* Allow particle effects to actually be unloaded.
* Implement the **forceStop** parameter to **ParticleEffect::Stop**. It can be used to immediately stop a particle effect.
* Implement **GlobalSettings::Save** as a stopgap.
* Minor API fixes

**Version 1.0.1  
August 31, 2022**

* Add the ability to remove UI gadgets.
* Fix an issue with Texture loading in the Visual Studio project template.
* Fix an internal method of FontManager accidentally being made public.

**Version 1.0.0  
August 30, 2022**

Initial release of the Lightning game engine.