Thank you for using THOR_Thunderstorm. Please don‘t forget to rate the product on the Unity Asset Store and/or leave a review.

Features:

- **Automatic spawning of lightning**
- **Lightning Bolts**
- **Sheet Lightning**
- **Lightning Evolution**
- **Dynamic Spotlight with Shadows** (optional)
- **26 High Quality Thunder samples in 4 distance categories** (Very Near, Near, Medium, Far)
- **3 Background Thunder Loops**
- **Automatic Pooling**
- **Zero Garbage Collection**
- **Very low Overdraw**
- **High compatibility**

Key aspects of THOR_Thunderstorm:

The goal is to provide a simple solution to add dynamic thunder and lightning to your scenes in an automated way. THOR Thunderstorm has been designed with 4 aspects in mind:

- **Performance**

THOR_Thunderstorm is very fast because it uses - simple shaders, very few textures, pooling, a minimum number of audiosources, only 1 spotlight.

- **Simplicity**

THOR_Thunderstorm is very easy to use. You are ready to use it by just adding the THOR_Thunderstorm prefab to your scene. The tidy custom inspector exposes only what you need. If you need more control you can go into the advanced settings. Every property in the inspector has useful tooltips.

- **Visual Fidelity**

THOR_Thunderstorm looks fantastic, even in slow motion. High speed video footage of lightning strikes was the reference. The scripts, shaders, textures and meshes were created to work well together for a truly immersive thunderstorm experience.

- **Audio Fidelity**

The 29 included samples are carefully recorded and edited with game audio in mind from the get go. Rain noise has been minimized during recording and carefully filtered out in editing. What you get is pure thunder and lightning ready for integration in your weather system.

Getting started:

Drag and drop the THOR_Thunderstorm Prefab into your scene and adjust the probability to your liking. By default THOR_Thunderstorm is configured to spawn lightning in a range of 300 - 5000 meters from the camera and at a height of 600 Meters. You may need to increase the Far clip plane of your camera to see the distant lightning(alternatively the THOR_Thunderstorm distance ranges can be adjusted, but a view distance of 5000 or more is recommended).

The API:

On awake a static instance is created for THOR_Thunderstorm. This allows you easy access from other scripts. THOR_Thunderstorm uses the namespace "THOR". Add "using THOR;" to the beginning of your script for simplicity.

If you want to set the probability from a script you can simply call:
THOR_Thunderstorm.SetProbability(yourFloatValue);

There is also a lerp function which allows you to set the probability over some time:
ControlThunderstorm(float targetIntensity, float transitionDuration);

If you want to access the probability directly you can use:
THOR_Thunderstorm.instance.probability = yourFloatValue;

Tweaks and Adjustments

Most of the settings are self explanatory and you can refer to the tooltips in the inspector, but there might be a few things that may not be very intuitive at first. The things that might need further explanation will be described below.

Changing the distances:

- If you want to increase or decrease the distances you have follow these steps:
- Check the "Show Advanced Settings" toggle.
- Open the "Distance & Duration Settings" foldout.
- Change the "Minimum Distance" and "Maximum Distance" to your liking. Note that the minimum distance should not be set too low or it can reveal that the lightning is a billboard like mesh.
- Adjust the "Brightness over Distance Lightning" and "Brightness over Distance Clouds" curves according to your distance ranges.
- (Optional) You may also want to change the "Maximum Distance", "Light Range" and "Intensity over Distance" in the "Light Settings".
- (Optional) You may also want to change the distance ranges as well as the "Volume over Distance" in the "Audio Settings".

Optimizing the number of Audiosources:

Under the "Audio Settings" you find the "Sound Duration Fade" curve, it controls how quickly a thunder sound fades out after the thunder sound started playing. By bringing the value in the curve to zero earlier, the thunder sound fades out quicker and the audio source becomes inactive sooner. With many layered thunder sounds shortening is not very noticeable and it might be a good optimization. To make this even less noticeable, THOR_Thunderstorm uses a background thunder loop sound that fades in and out after a thunder sound was playing. This loop consists of low thunder rumble that can usually be heard at the end of a thunder. If you change the "Sound Duration Fade" you may also want to adjust the "Fade In Time" and the "Fade Out Time" of the background loop sound.

Another way to reduce the number of audio sources is to reduce the "Far Distance" value. It controls the distance up to which a thunder sound will be played.

Note that an active object in the pool does not necessarily mean an active audio source, since the object can be waiting for the speed of sound delay. In this case the object does not cost performance in any way.

Pooling Considerations:

Since every lightning that lies within “Far Distance“ in the “Audio Settings“ has its own audio source that may keep the lightning object active in the pool, you have to keep in mind that changing the following values may need adjustments of the “Pool Size Lightning“ in the “Pool Settings“.

“Far Distance“ in the “Audio Settings“
“Sound Duration Fade“ in the “Audio Settings“
“Speed of Sound“ in the “Audio Settings“

To check if you use the pooling efficiently, select the THOR_Thunderstorm object in the hierarchy and inspect its children. Ideally there stays one “THOR_Lightning(Clone)“ child inactive at all times but not more. if you see all the children are active, it could mean that your pool size is too small. If more children are inactive at all times, the pool might be too big. Make such tests at the maximum lightning probability that you intend to use.

The Flicker Curves

The way the flickering works is that a lightning randomly samples a section of the flicker curve over its duration (that‘s why the flicker curve must be set to loop). There are 2 separate flicker curves, one for Lightning Bolts and one for the Lightning Clouds. It is recommended to make the Lightning Clouds Flicker Curve use a copy of the Lightning Bolts Curve and then lower its value range. This way the flickering stays in sync.

Optimizing for Mobile

Minimize the number of audio sources as described above and disable the “Enable Light“ Checkbox in the “Light Settings“. You may also want to lower the distances as described above.

Depth Blending

Depth Blending is a feature that is also used a lot with particles (called smooth particles in Unity). It works the same way in THOR_Thunderstorm and also has the same requirement. To use it, you need to make your camera render a depth texture. In the deferred rendering path this happens automatically, in forward rendering you need to instruct the camera via script to do so. The code to enable it is: “Camera.main.depthTextureMode = DepthTextureMode.Depth;“

Final Words:

Have a fun time and big success using THOR_Thunderstorm, please remember you can support this tool by leaving a review for this product on the Unity Asset Store.

May Thor be with you!