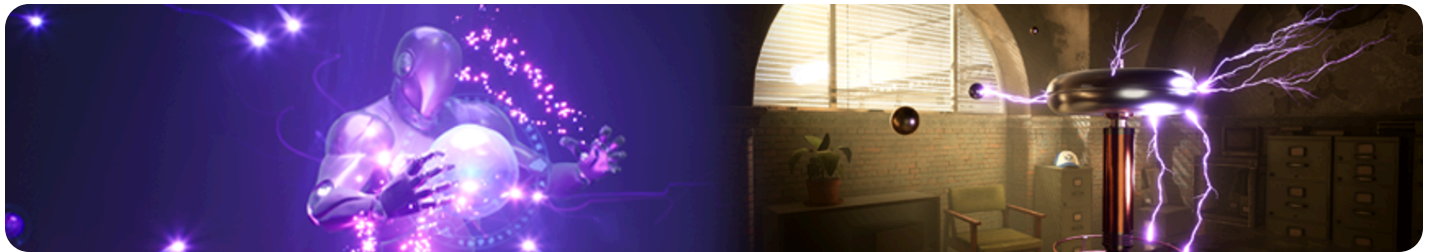


# System Spawn Group

This document provides reference information for modules in the System Spawn group.



**System Spawn** modules occur once per created system. Modules in this section set up initial values for each system. Modules are executed in order from the top to the bottom of the stack.

Each of the module types in the System Spawn group has its own section in this document, with tables describe how that module functions. Keep in mind that you can create custom modules for any part of a Niagara system or emitter. The ones listed here are just the ones that are automatically included with Unreal Engine.

## Spawning Modules

Module	Description
<b>Spawn Burst Instantaneous</b>	This module causes all emitters to spawn a burst of particles spontaneously.

# New Scratch Pad Module

Selecting this item in the **Add** (Plus sign) menu opens the **Scratch Pad** panel (by default this docks next to the **System Overview**) and places a **Scratch Pad module** in the **Selection** panel. You can also open the Scratch Pad panel by using **Windows > Scratch Pad**. However, by placing a Scratch Pad module in the stack, any modules or dynamic inputs you create in the Scratch Pad are automatically connected to your script. If you open the Scratch Pad panel using the Windows menu, any items you create there will have to be added to your script manually.

## Set New or Existing Value Directly

Selecting this item from the **Add** (Plus sign icon) menu places a **Set Parameter** module in the **Selection** panel. Click the **Plus sign (+)** icon to select **Add Parameter** or **Create New Parameter**. Choices for each of these are in the sections below.

## Add Parameter

When you select **Set Specific Parameter**, you select from the parameters listed. This adds a **Set Parameter** module to the System Spawn group.



Some of these parameters can be set or modified in other modules. Some are only set using a Set Variable module.

Parameter	Description
<b>System.Age</b>	This parameter defines the age of the named system.
<b>System.LoopedAge</b>	This parameter calculates the age of the system relative to its current loop. For example, if a system has been active for <b>8</b> seconds and it loops every <b>5</b> seconds, the system's LoopedAge will be <b>3</b> seconds. LoopedAge returns to 0 every time a system loops.

Parameter	Description
<b>System.NormalizedLoopAge</b>	This parameter calculates the age of the system relative to its current loop, normalized from <b>0</b> to <b>1</b> . NormalizedLoopAge is expressed as <b>LoopedAge</b> divided by <b>CurrentLoopDuration</b> . If a system has been active for <b>8</b> seconds and it loops every <b>5</b> seconds, the system's LoopedAge will be <b>3</b> . The system's NormalizedLoopAge will be <b>0.6</b> .
<b>System.CurrentLoopDelay</b>	This parameter defines the current amount of delay before the named system's current loop repeats.
<b>System.CurrentLoopDuration</b>	This parameter defines the duration of the named system's loop.
<b>System.ExecutionState</b>	This affects the state of the system. Valid value choices are: <ul style="list-style-type: none"> <li>• Active</li> <li>• Inactive</li> <li>• InactiveClear</li> <li>• Complete</li> </ul>
<b>System.ExecutionStateSource</b>	This indicates the source of an execution state setting. It is used to allow scalability to change the state, but only if the state has not been defined by something with higher precedence.
<b>System.LocalSpace</b>	This parameter defines whether the position of particles is respective to the world origin or the owning Niagara Component's location. Settings are: <ul style="list-style-type: none"> <li>• <b>False:</b> Particle position is in WorldSpace and will be relative to the World origin. A particle with position <b>0,0,0</b> will render at the World origin.</li> <li>• <b>True:</b> Particle position is in LocalSpace and will be relative to the owning Niagara Component's location. A particle with position <b>0,0,0</b> will render at the owning Niagara Component's location.</li> </ul>

Parameter	Description
<b>System.LoopCount</b>	This parameter defines how many times the system's loop repeats.
<b>System.ExecutionState</b>	<p>This affects the state of the system. Valid value choices are:</p> <ul style="list-style-type: none"> <li>• <b>Active</b></li> <li>• <b>Inactive</b></li> <li>• <b>InactiveClear</b></li> <li>• <b>Complete</b></li> </ul>
<b>System.ExecutionStateSource</b>	This variable is linked to the ENiagaraExecutionStateSource parameter, which indicates the source of a system execution state setting. It is used to allow scalability to change the state, but only if the state has not been defined by something with higher precedence.

## Create New Parameter



When you select **Create New Parameter**, you select from the parameters listed. This adds that parameter to the **Set Parameter** module in the System Spawn group.

Parameter	Type	Description
<b>Audio Oscilloscope</b>	<b>Data interface</b>	<p>This adds a new Audio Oscilloscope data interface module to the emitter.</p> <p>The Audio Oscilloscope module can directly access the waveform data of the audio signal.</p>
<b>Audio Spectrum</b>	<b>Data interface</b>	<p>This adds a new Audio Spectrum data interface module to the emitter. The Audio Spectrum module can drive a</p>

Parameter	Type	Description
		visualization based on how loud the audio is at specific frequencies.
<b>Bool</b>	<b>Primitive</b>	This adds a Set Variable module that has a true/false checkbox.
<b>Camera Query</b>	<b>Data Interface</b>	This adds a new Camera Query data interface module to the emitter. This data interface can be used to retrieve camera information (camera position, rotation, FOV, etc) for the specified controller index.
<b>ENiagaraBooleanLogicOps</b>	<b>Enum</b>	<p>This is an enumeration used by various modules and dynamic inputs that want to test using boolean logic:</p> <ul style="list-style-type: none"> <li>• <b>Greater Than</b></li> <li>• <b>Greater Than Or Equal To</b></li> <li>• <b>Equal To</b></li> <li>• <b>Not Equal To</b></li> </ul>
<b>ENiagaraCoordinateSpace</b>	<b>Enum</b>	<p>This is an enumeration used by various modules and dynamic inputs to distinguish between coordinate spaces:</p> <ul style="list-style-type: none"> <li>• <b>Simulation:</b> If the emitter is set to Local, use Local. Otherwise, use World.</li> <li>• <b>World:</b> In the world space of the game.</li> <li>• <b>Local:</b> In the coordinate space of the owning component.</li> </ul>
<b>ENiagaraExecutionState</b>	<b>Enum</b>	This enumeration type is used by parameters that manage system or emitter execution states, such as

Parameter	Type	Description
<b>Emitter.ExecutionState</b> or <b>System.ExecutionState.</b>		
<b>ENiagaraExecutionStateSource</b>	<b>Enum</b>	This indicates the source of an execution state setting. It is used to allow scalability to change the state, but only if the state has not been defined by something with higher precedence.
<b>ENiagaraExpansionMode</b>	<b>Enum</b>	<p>This enumeration is used by location modules to determine where the origin point of expansion is:</p> <ul style="list-style-type: none"> <li>• <b>Inside</b></li> <li>• <b>Centered</b></li> <li>• <b>Outside</b></li> </ul>
<b>ENiagaraLegacyTrailMode</b>	<b>Enum</b>	This enum controls the way that the width scale property affects animation trails. This is only used for Legacy Anim Trail support when converting from Cascade to Niagara.
<b>ENiagaraOrientationAxis</b>	<b>Enum</b>	<p>This is an enumeration used by several modules to determine which axis to do calculations with:</p> <ul style="list-style-type: none"> <li>• <b>X Axis</b></li> <li>• <b>Y Axis</b></li> <li>• <b>Z Axis</b></li> </ul>

Parameter	Type	Description
<b>ENiagaraRandomnessMode</b>	<b>Enum</b>	<p>This sets the type of random number generation used by this emitter. Valid choices are:</p> <ul style="list-style-type: none"> <li>• <b>Simulation Defaults</b></li> <li>• <b>Deterministic</b></li> <li>• <b>Non-Deterministic</b></li> </ul>
<b>Float</b>	<b>Primitive</b>	This creates a float value variable.
<b>Grid2D Collection</b>	<b>Data Interface</b>	This is used with simulation stages. It enables the user to read or write to a 2D array of data, and then iterate over each pixel in the grid during a simulation stage.
<b>Int32</b>	<b>Primitive</b>	This creates an integer variable.
<b>Linear Color</b>	<b>Primitive</b>	This creates an RGBA color variable, represented as a color picker.
<b>Matrix</b>	<b>Primitive</b>	This creates a 4×4 matrix variable.
<b>Mesh Tri Coordinate</b>	<b>Struct</b>	This is a simple struct containing a triangle index along with a barycentric coordinate on the face of that triangle.
<b>Neighbor Grid 3D</b>	<b>Data Interface</b>	This is used with simulation stages. It enables the user to read or write to a 3D array of data, and then iterate over each pixel in the volume during a simulation stage.
<b>Niagara ID</b>	<b>Struct</b>	This is a two-part struct used to track particles. It allows fast access to this particle's data. It is always unique

Parameter	Type	Description
		<p>among currently living particles, but will be reused after the particle dies.</p> <p><b>AcquireTag</b> is a unique tag for when this ID was acquired. It allows us to differentiate between particles when one dies and another particle reuses the dead particle's index.</p>
<b>Occlusion Query</b>	<b>Data Interface</b>	<p>This adds a new Occlusion Query data interface module to the emitter. The data interface is used to read depth buffer occlusion information.</p> <div>  <p>This can only be used with GPU emitters.</p> </div>
<b>Quat</b>	<b>Primitive</b>	<p>This creates a quaternion variable, used to represent rotations.</p>
<b>Simple Counter</b>	<b>Data Interface</b>	<p>This adds a new Simple Counter data interface module to the emitter. This data interface enables you to increment a thread-safe counter.</p> <div>  <p>This can only be used with CPU emitters.</p> </div>
<b>Particle Attribute Reader</b>	<b>Data Interface</b>	<p>This adds a new Particle Attribute Reader data interface to the emitter. The data interface can be used to query particle payload values from other emitters, and can sometimes be easier to use than Events.</p>



Parameter	Type	Description
<b>Spawn Info</b>	<b>Struct</b>	This is a structure used in spawning to specify the <b>Count</b> of particles to create, <b>InterpStartDt</b> or offset from the current frame begin time to start spawning, <b>IntervalDt</b> defining the time gap between particles being spawned, and <b>SpawnGroup</b> allowing spawned particles to belong to different categories.
<b>Vector</b>	<b>Primitive</b>	This creates a three-channel set of floats.
<b>Vector 2D</b>	<b>Primitive</b>	This creates a two-channel set of floats.
<b>Vector 4</b>	<b>Primitive</b>	This creates a four-channel set of floats.
<b>Vector Field</b>	<b>Data Interface</b>	This is a data interface with functions to query a vector field.
<b>Volume Texture Sample</b>	<b>Data Interface</b>	This adds a new Volume Texture data interface module to the emitter. You can use this to sample a volume texture.