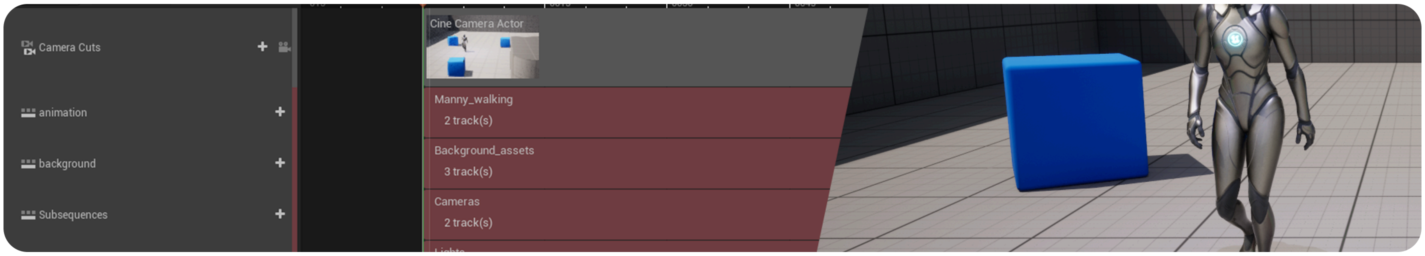
Developer
/ Documentation
/ Unreal Engine
/ Unreal Engine 5.4 Documentation
/ Animating Characters and Objects
/ Cinematics and Sequencer
/ Sequencer Overview
/ Spawnables and Possessables

Spawnables and Possessables

Spawn temporary Actors, lights, and other objects in your scene by using Spawnables.



In Sequencer, you can choose to either reference an Actor that currently exists in the scene, referred to as a **Possessable**, or spawn a new Actor, referred to as a **Spawnable**. This document provides an overview of these concepts and how you can use them in your scenes.

Prerequisites

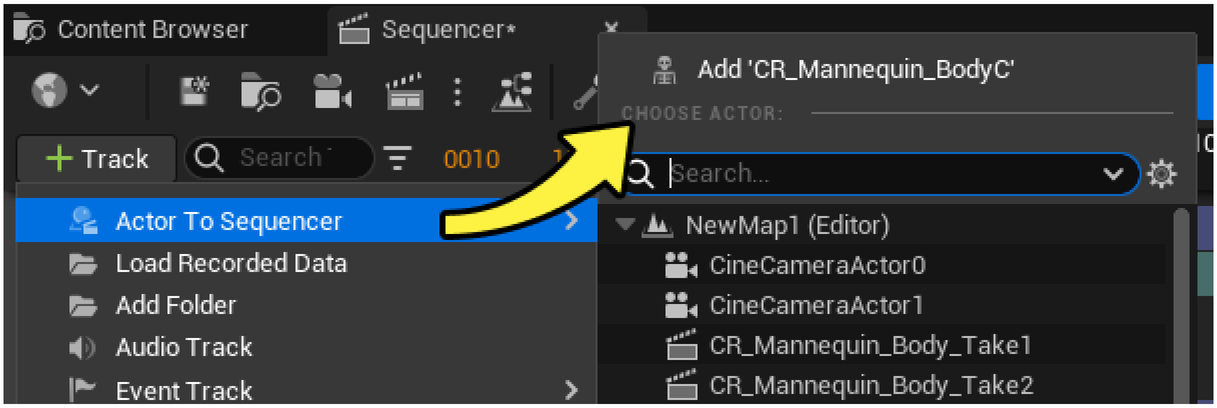
- You have an understanding of **Sequencer** and its **Interface**.
- You know how to create and use **Tracks**.

Possessable and Spawnables

Possessable

You can possess Actors by adding existing Actors in a level to your sequence. The link is formed as a soft object path. In most cases this might be an acceptable or even preferable workflow. If your scene requires a high degree of interaction with currently existing Actors in a **Level Sequence**, then possessing is likely the best choice for referencing Actors.

You can add any Actor to the sequence by selecting it or picking an Actor from the **Add Actor to Sequencer** list. You can also drag an Actor from the **Outliner** to an open space in **Sequencer**.



Spawnable

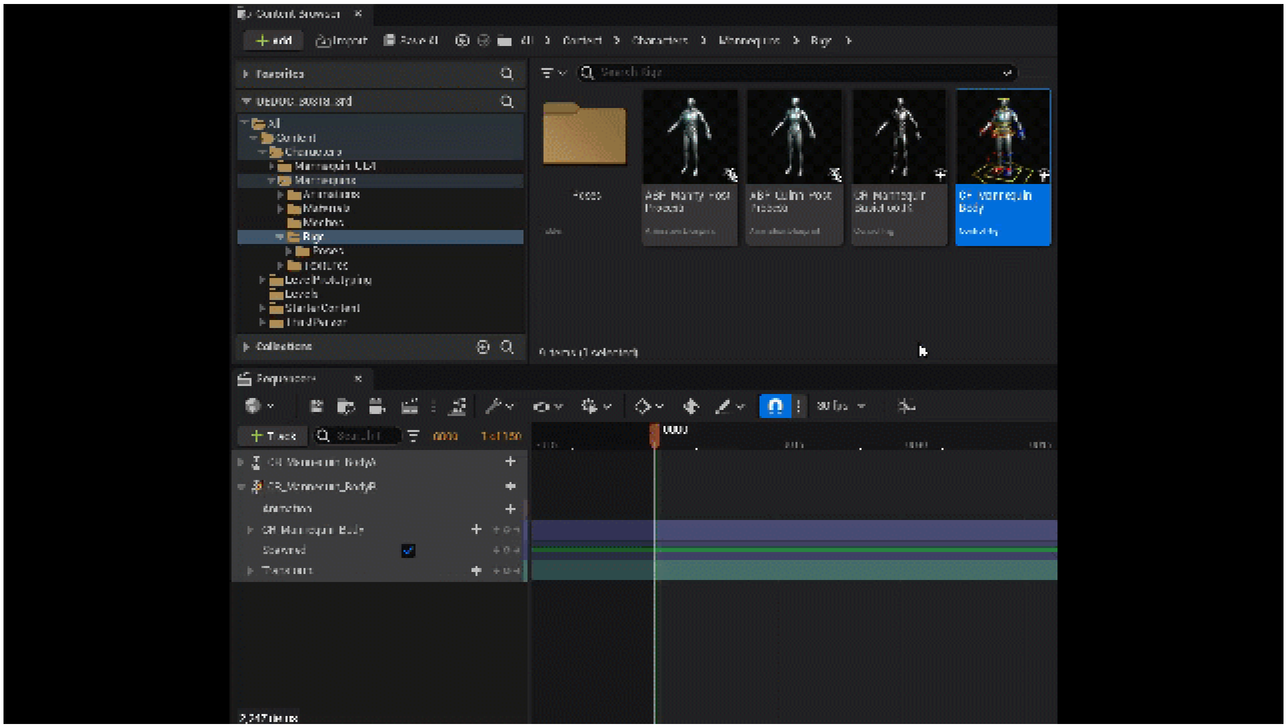
In cases where your scene requires Actors that might be temporary for the duration of a scene, you can use **Spawnables**. By default, when a sequence with a Spawnable Actor starts, the Actor will be spawned. When the sequence ends, the Actor will be destroyed and removed. You can also explicitly control the frames during which the Actor is spawned and destroyed with the Spawn Track.

Create Spawnable

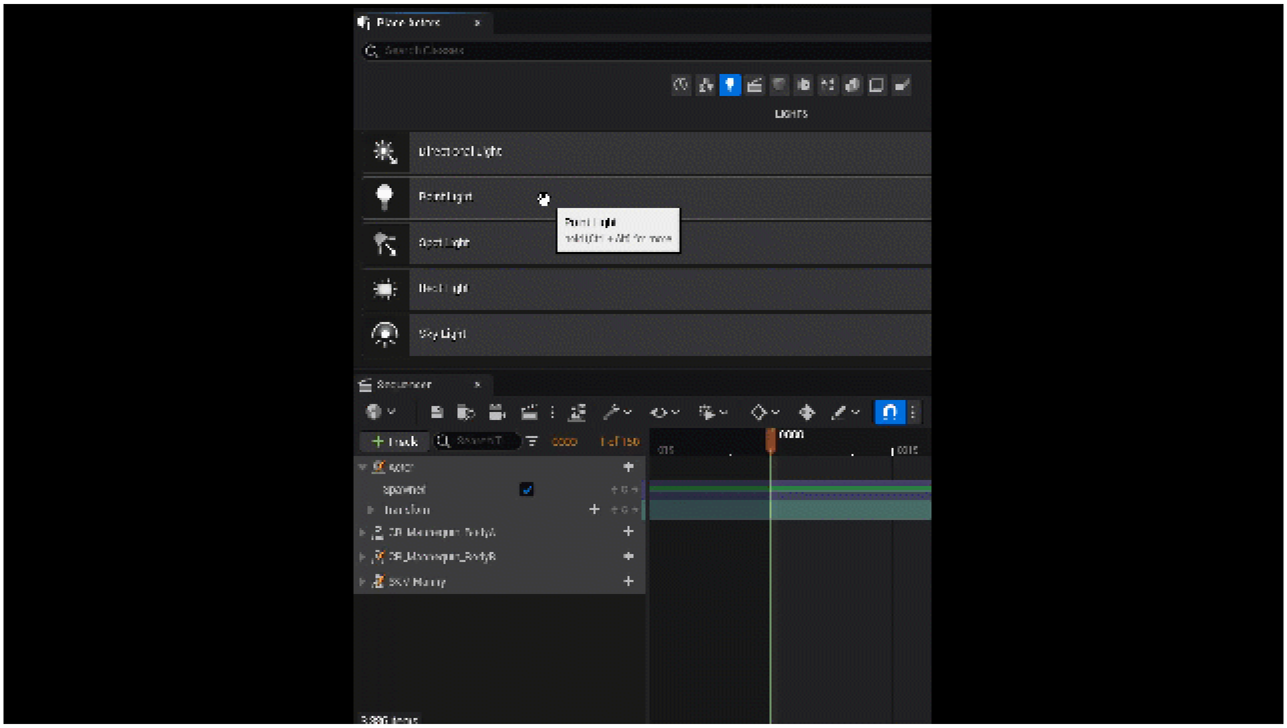
There are two methods for spawning Actors: dragging actors and adding an Actor.

If you have an existing Actor that isn't in **Sequencer** and you want to turn it into a **Spawnable**, add it to **Sequencer** as a **Possessable** and then convert the Actor into a **Spawnable**.

You can create a **Spawnable** by dragging it into Sequencer. Drag an Actor from the **Content Browser**, **Outliner**, or **Place Actors** panels into an open area in Sequencer Outliner.



Drag actor from Content Browser to Sequencer



Drag actor from Place Actors to Sequencer

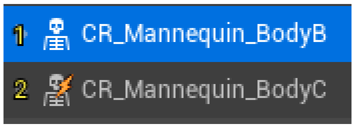
To convert a **Possessable** to a **Spawnable**, follow these steps:

1. Select the Actor that you want to spawn in the Viewport or from the Outliner.
2. In Sequencer, click **Track > Add Actor to Sequencer > Add '{name of selected actor}'**. This action adds the Actor to Sequencer but it is not a Spawnable yet.
3. Right-click the Actor in Sequencer and select **Convert to Spawnable**.

A Spawnable Actor can be converted back to Possessable. When this happens, the Actor is re-created in your level and the track binds to it, removing the Spawnable Actor.

Identify Spawnables in Sequencer

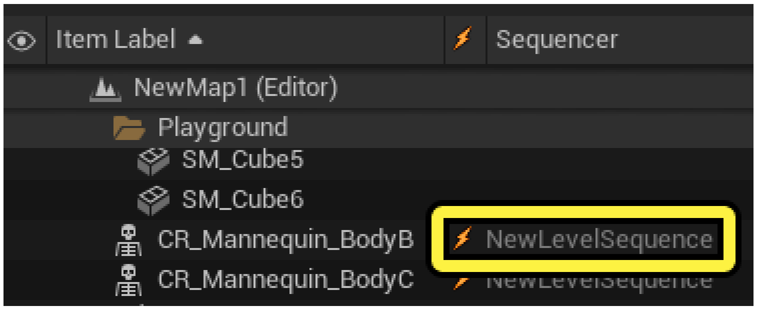
Once an Actor has been spawned, a **lightning bolt overlay** appears on the Actor in Sequencer and in the **Outliner**.



1 CR_Mannequin_BodyB

2 CR_Mannequin_BodyC

Possessable Actor Spawnable Actor



Item Label ▲	⚡ Sequencer
NewMap1 (Editor)	
Playground	
SM_Cube5	
SM_Cube6	
CR_Mannequin_BodyB	⚡ NewLevelSequence
CR_Mannequin_BodyC	⚡ NewLevelSequence

Spawned actors in Outliner

Spawnable Properties

Spawnable Actors have various properties that control their behavior and interactions with Sequencer. These properties can be accessed by right-clicking the Spawnable bound track and locating the **Spawnable** category.

Property Name	Description
Spawned Object Owner	Specifies which Level Sequence owns the Actor, which also determines the automatic spawn behavior. • **This Sequence** is the default setting which causes the Actor to be spawned and destroyed for the duration of the current sequence only.

Property Name	Description
	- **Root Sequence** causes the Actor to spawn and despawn for the duration of the master sequence, if one is being used. This causes the Actor to be spawned and destroyed outside the bounds of the current sequence. - **External** causes the Actor to spawn at the start of the sequence, but it will not be destroyed at the end. You can instead destroy the Actor through Blueprints by using **Sequencer Tags and Groups**.
Spawnable Level	Specifies which Level the Actor will spawn into. This list is determined by what Levels exist in your **Levels** window. ⓘ You cannot specify the World Partition or Data Layers.
Change Class	This option provides a way to change which class is being spawned, while preserving any Sequencer tracks already added to this Spawnable. This does not preserve any non-Sequencer data such as object properties.
Continuously Respawn	When enabled, the Actor is checked every tick to ensure that it still exists (based on the status of the Spawn track). The Actor respawns in the event that an external destroy event did destroy it.
Evaluate Tracks When Not Spawned	When enabled, all tracks from this Actor will still be evaluated even if the Actor is not spawned. This is useful if the Actor requires any preprocessing before being spawned.
Net Addressable	When enabled, this Spawnable Actor will be spawned using a unique name that allows it to be referenced by the server and client.

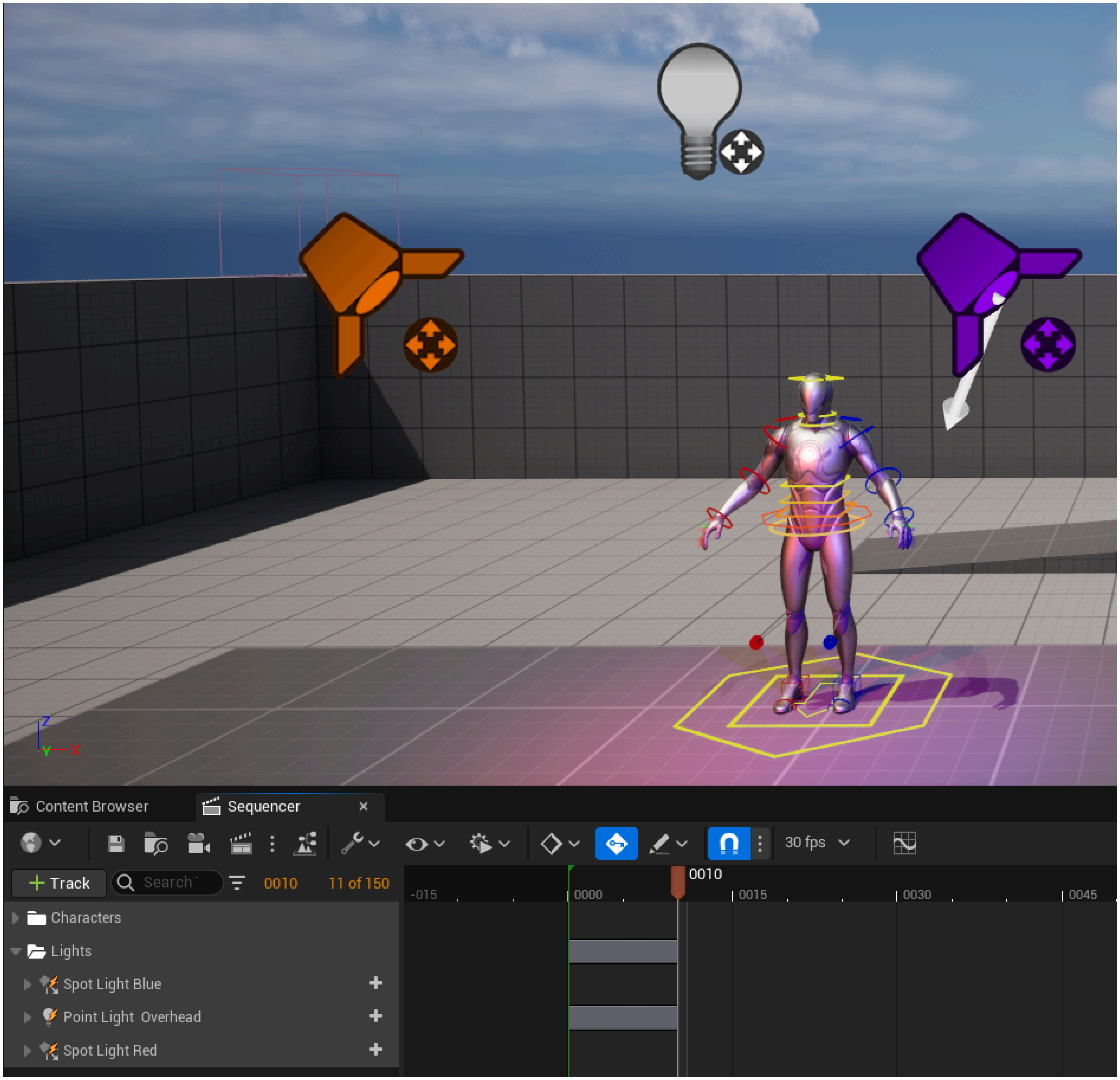
Property Name	Description
Save Default State	Saves the current state of this Spawnable Actor. Typically, you don't need to click this as Unreal Engine will attempt to automatically save any changes to the Spawnable Actor.
Convert to Possessable	Converts the Actor to a Possessable Actor. When this occurs, the Actor is re-created in your Level and the Actor track binds to it, removing the Spawnable Actor.

Workflows

Spawnable Actors helps you create more organized scenes in terms of content organization and management.

Lighting Scenes

Instead of placing several lights in your Level that have to be manually enabled or disabled per shot, you can add the required lights to your shot as Spawnable Actors. They will exist only for that shot and don't clutter your Level with unnecessary lights.



This workflow can also be used for any temporary Actor, such as particles so that it can create an instance of it.