



UNREAL  
ENGINE

HOUR 15

Level Blueprint:  
Creating Level Events

# INTRODUCTION

---

In this lecture, you will use the Level Blueprint to learn basic scripting concepts, such as how to assign an Actor to an event. You will also learn how to assign Actors to reference variables and change their properties through Blueprint.



# LECTURE GOALS AND OUTCOMES

## Goals

---

The goals of this lecture are to

- Learn to assign Actors to events
- Learn to assign Actors to reference variables
- Learn to change the properties of an Actor through Blueprint

## Outcomes

---

By the end of this lecture you will be able to

- Assign an Actor to an event
- Assign an Actor to a reference variable
- Change the properties of an Actor based on player actions



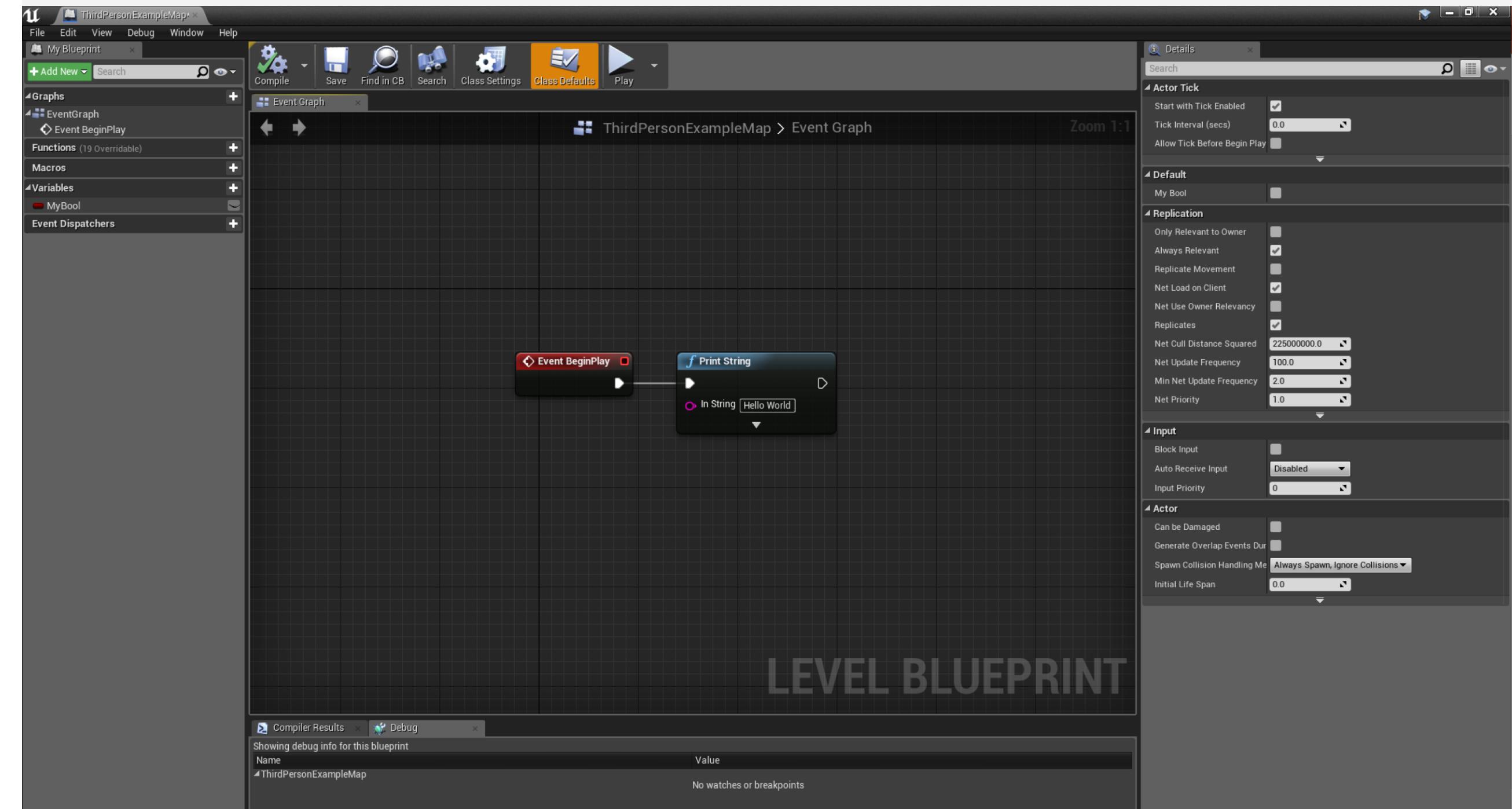
# LEVEL BLUEPRINTS

Basic Concepts



# LEVEL BLUEPRINT

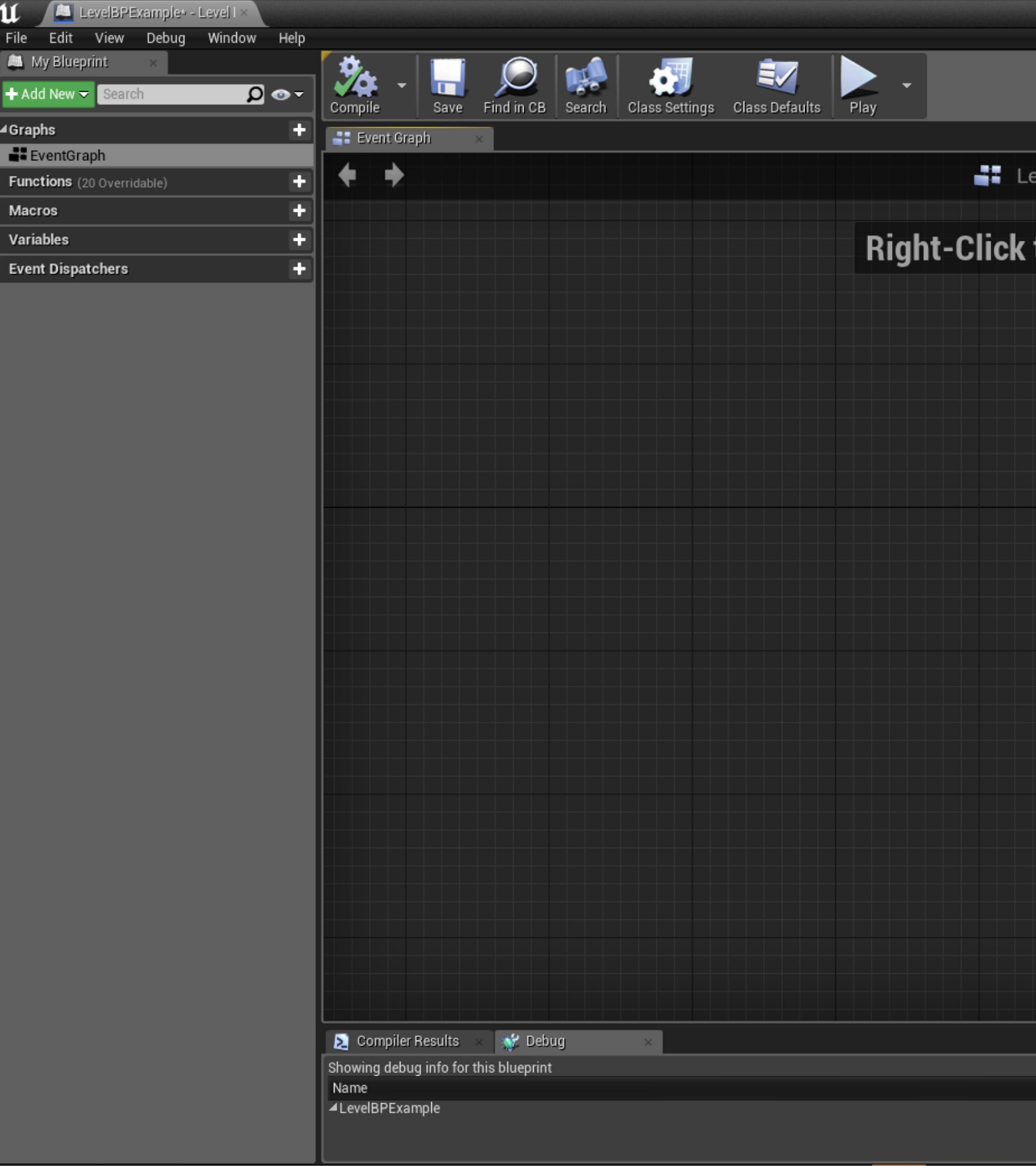
A Level Blueprint is a specialized type of Blueprint that acts as a Level-wide global event graph. Each Level in your project has its own Level Blueprint created by default.





# LEVEL BLUEPRINT

Level Blueprints are used to drive Level-based events and control internally scripted behaviors for in-game Actors placed in the Level.





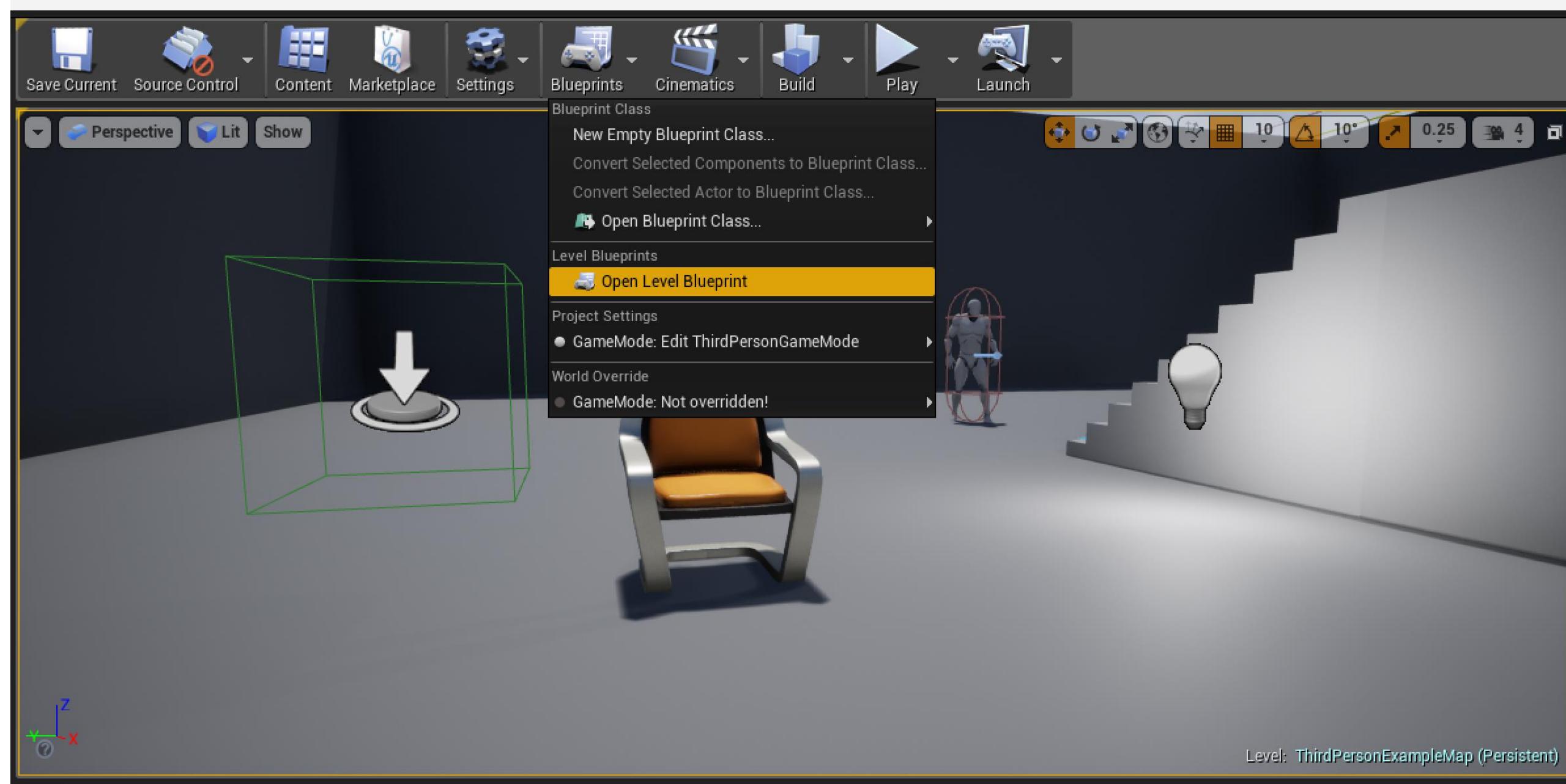
**Each Level created  
has a Level Blueprint  
associated with it.**

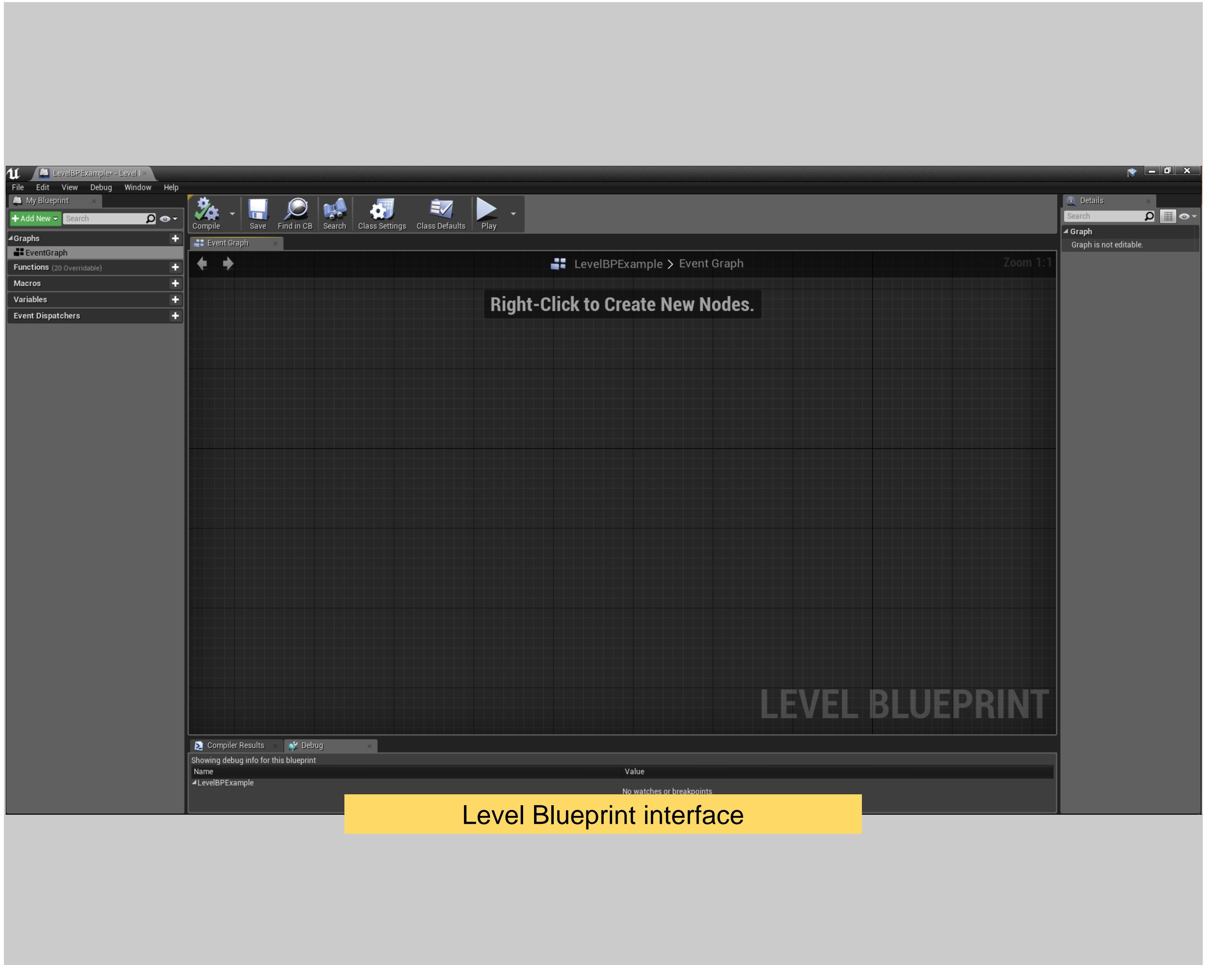
Although the Level Blueprint is associated with a Level, it does not know about the Actors in the Level unless you tell it about them. You can do this by assigning an Actor to either an event or an Actor reference variable.



# OPENING A LEVEL BLUEPRINT

To open the Level Blueprint for the current Level, click on the Blueprint icon and in the drop-down select Open Level Blueprint.





## LEVEL BLUEPRINT INTERFACE

The Level Blueprint Editor is similar to other Blueprint Editors except that it does not have a Components tab, a Viewport tab, or a Construction Script function.



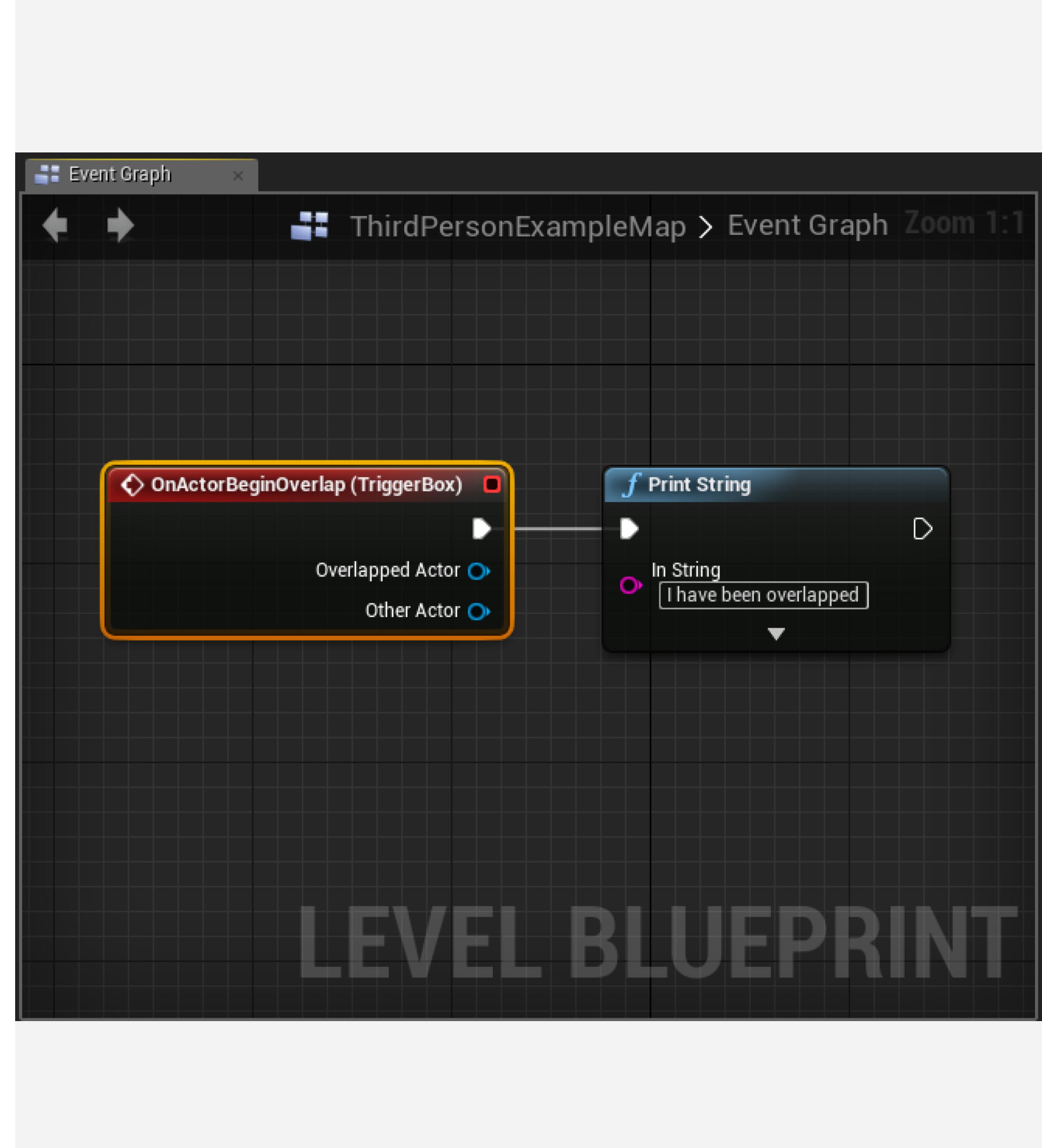
# LEVEL BLUEPRINT

Assigning Actors to Events



## ASSIGNING AN ACTOR TO AN EVENT

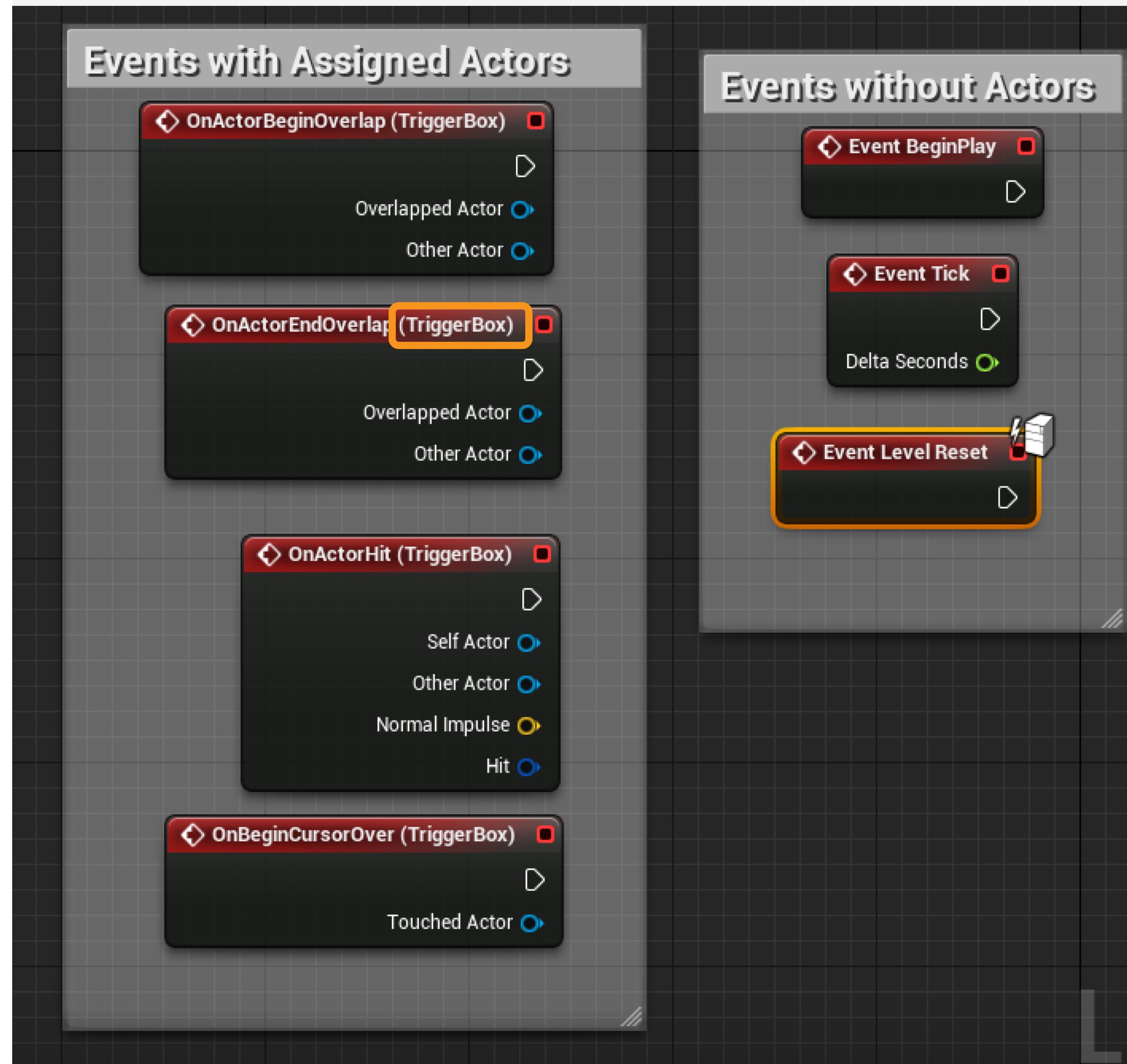
Blueprints are event based, meaning that an event needs to happen in order for a Blueprint sequence to be executed. In the Level Blueprint, events typically need an Actor to be assigned to them.





# ASSIGNING AN ACTOR TO AN EVENT

On the right are examples of common events. Some events need assigned Actors, and some do not. Events with assigned Actors show the name of the Actor that has been assigned to the event. In this case, the Actor is called TriggerBox.



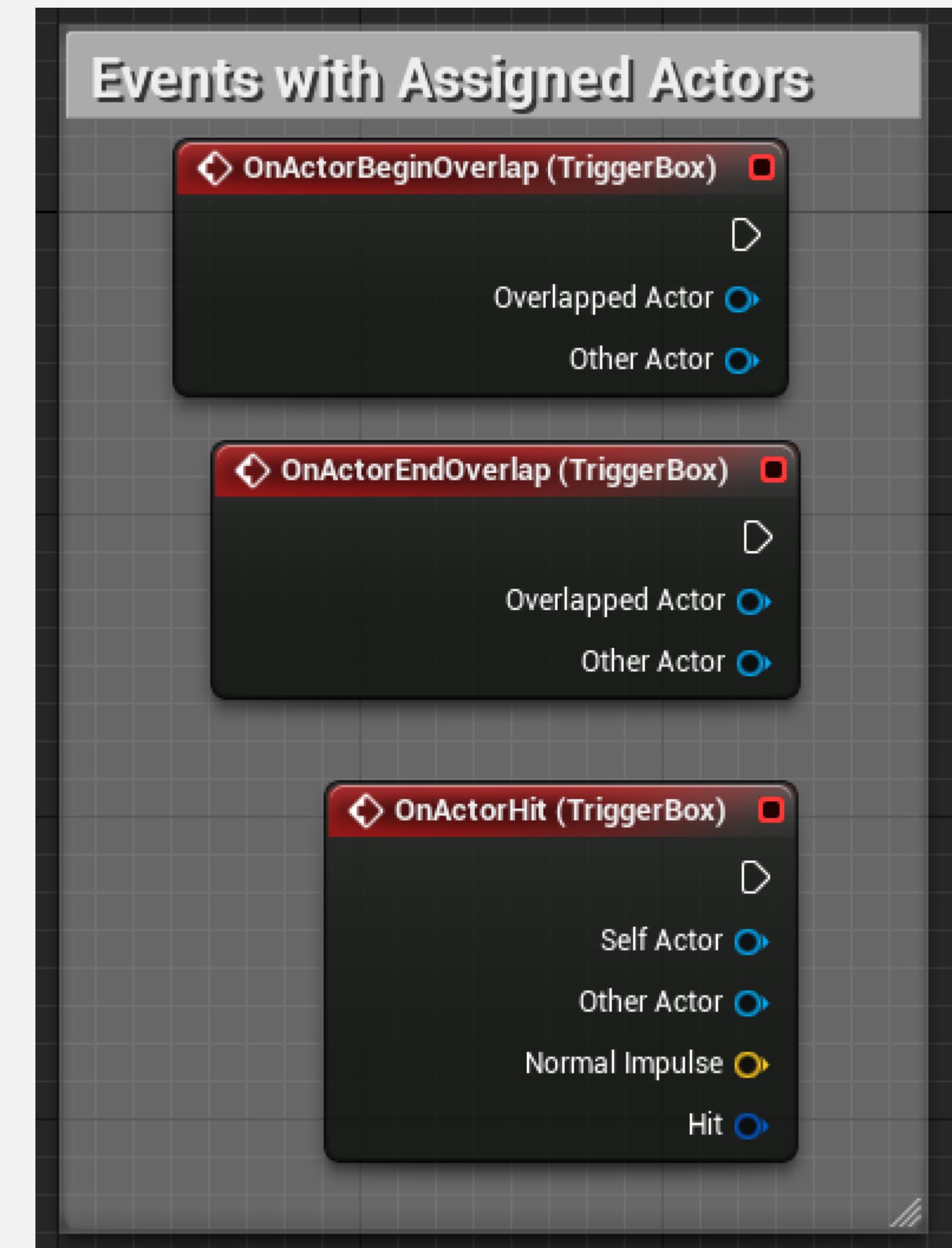
# LEVEL BLUEPRINT

Collision Events



## COLLISION EVENTS

Some of the most common types of events are collision events. Collision events require an Actor to be assigned to them, and they fall into two categories: Overlap and Hit events.

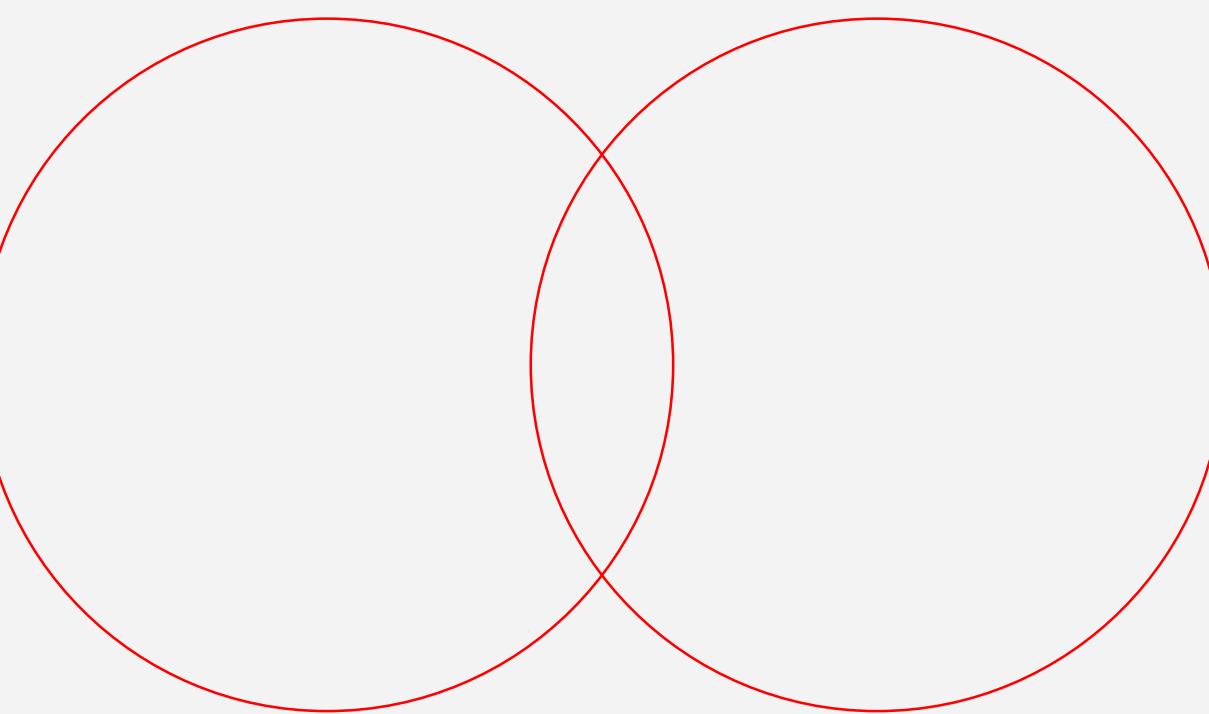




## OVERLAP EVENTS

---

Overlap events happen when a collision hull of an Actor or its components intersects with the collision hull of another Actor or its components.





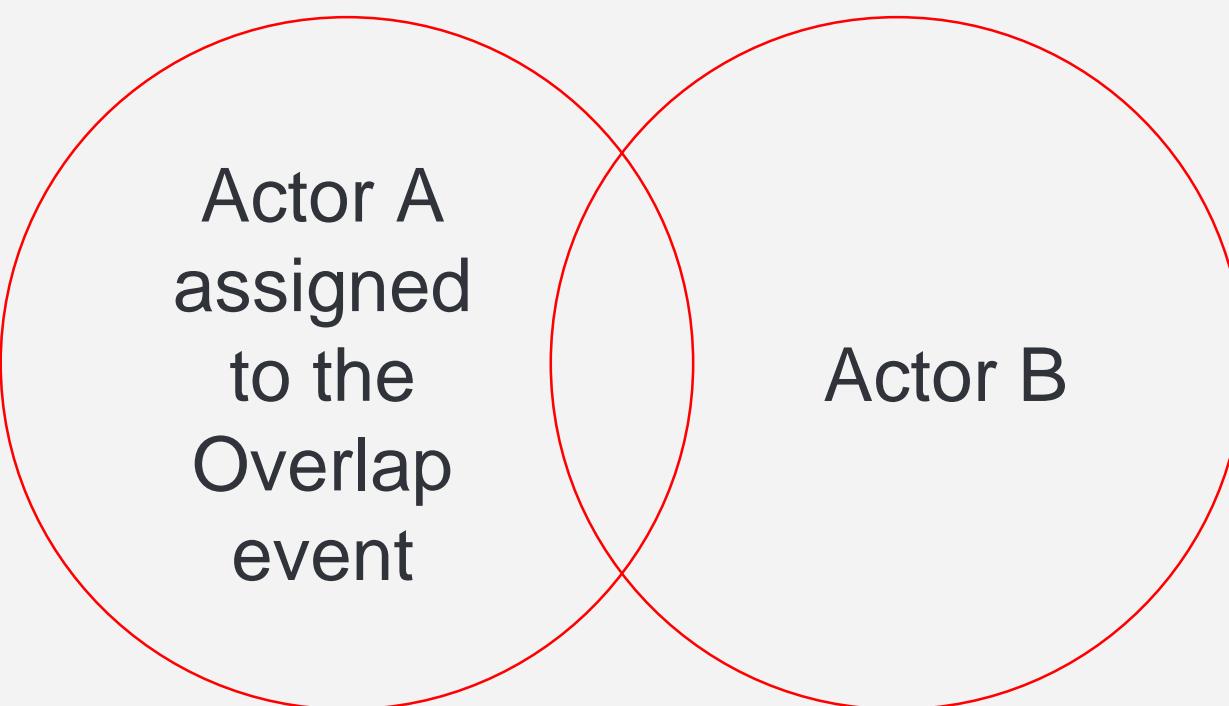
## OVERLAP EVENTS

---

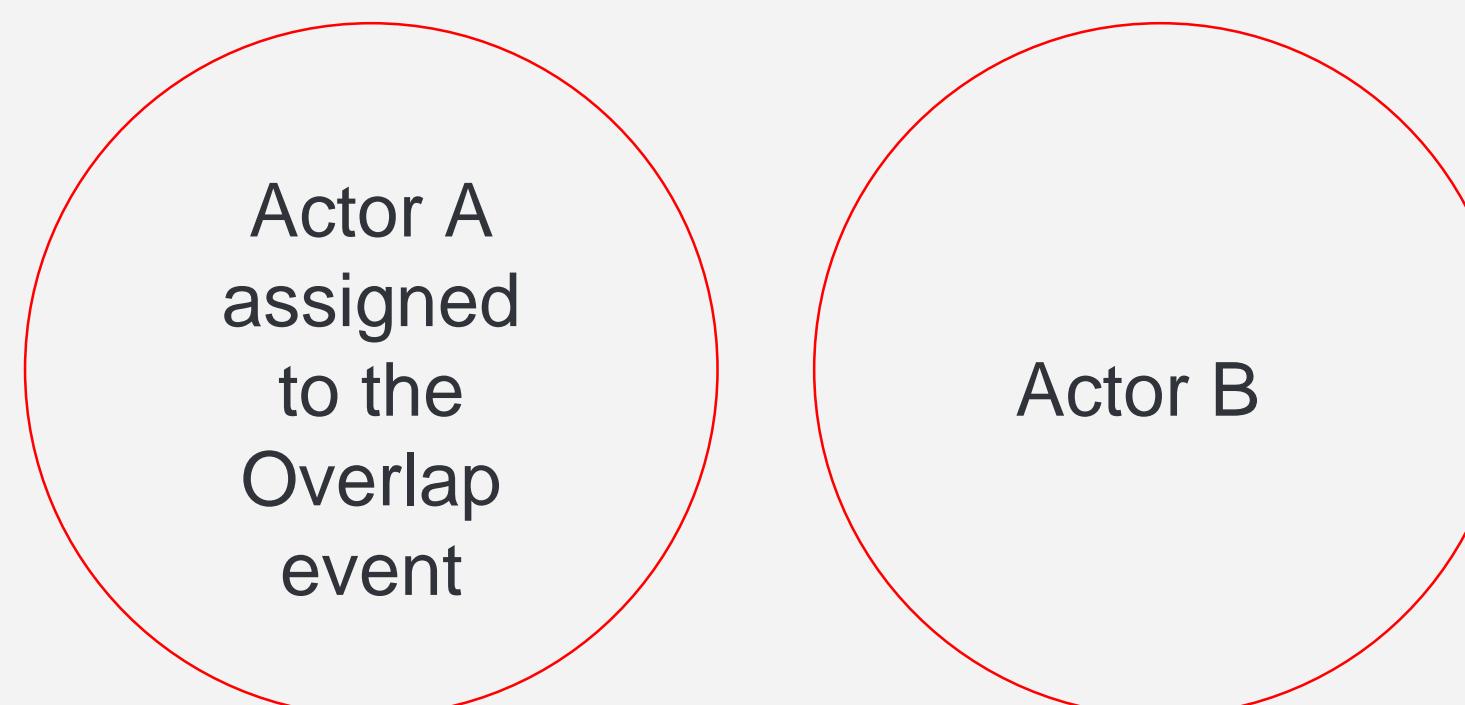
Overlap events are then broken down into two states. When collision hulls first intersect, the event is referred to as a Begin Overlap event; when they stop overlapping, it's called an End Overlap event.

Both Overlap event types execute once each time the Actor assigned to the event is overlapped.

**Begin Overlap:**  
**Executes once, when collision hulls first overlap**



**End Overlap:**  
**Executes once, when collision hulls stop overlapping**



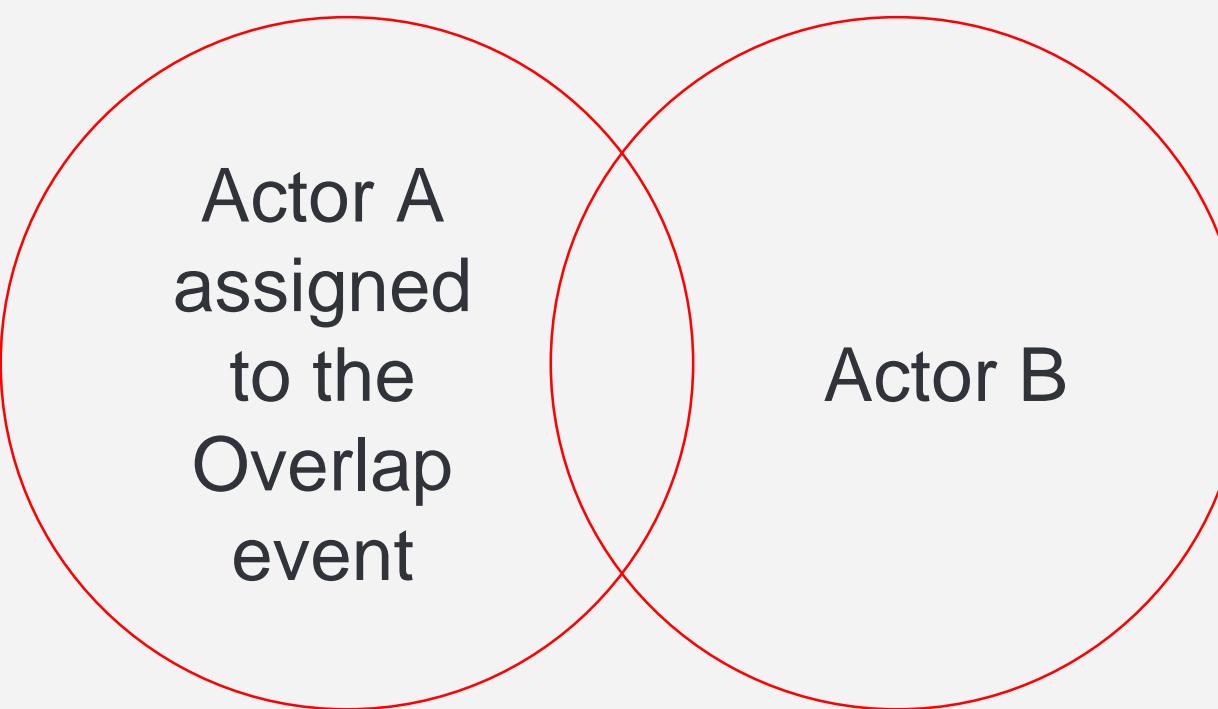


## OVERLAP EVENTS

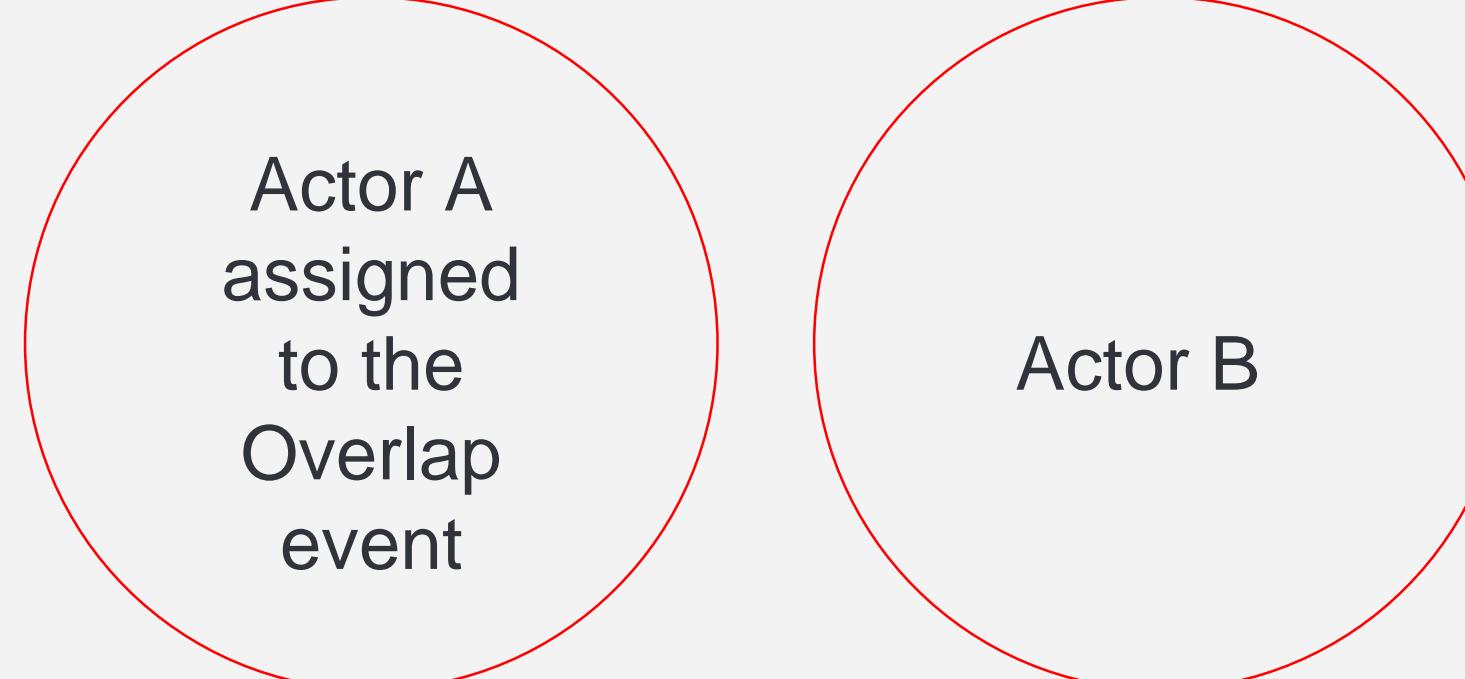
---

For example, if Actor A (assigned to the event) is overlapped by Actor B, the Begin Overlap event is executed. If another Actor overlaps Actor A, the Begin Overlap event will be executed again.

**Begin Overlap:**  
**Executes once, when collision hulls first overlap**



**End Overlap:**  
**Executes once, when collision hulls stop overlapping**



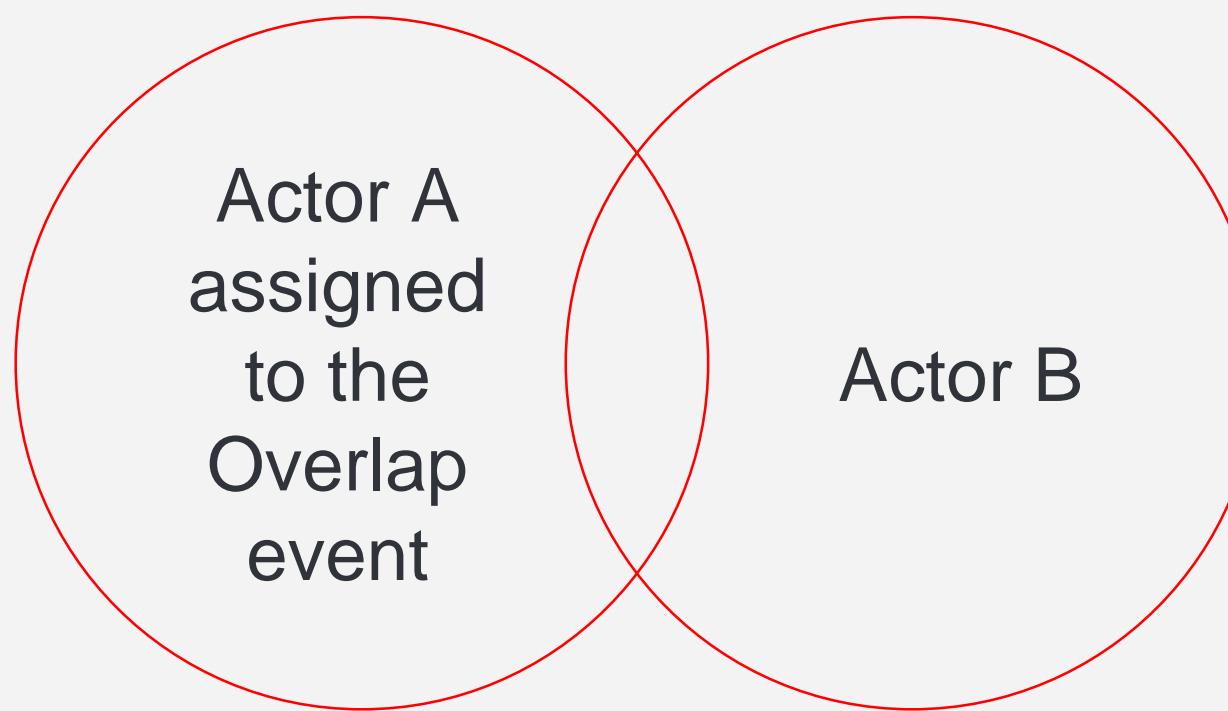


## OVERLAP EVENTS

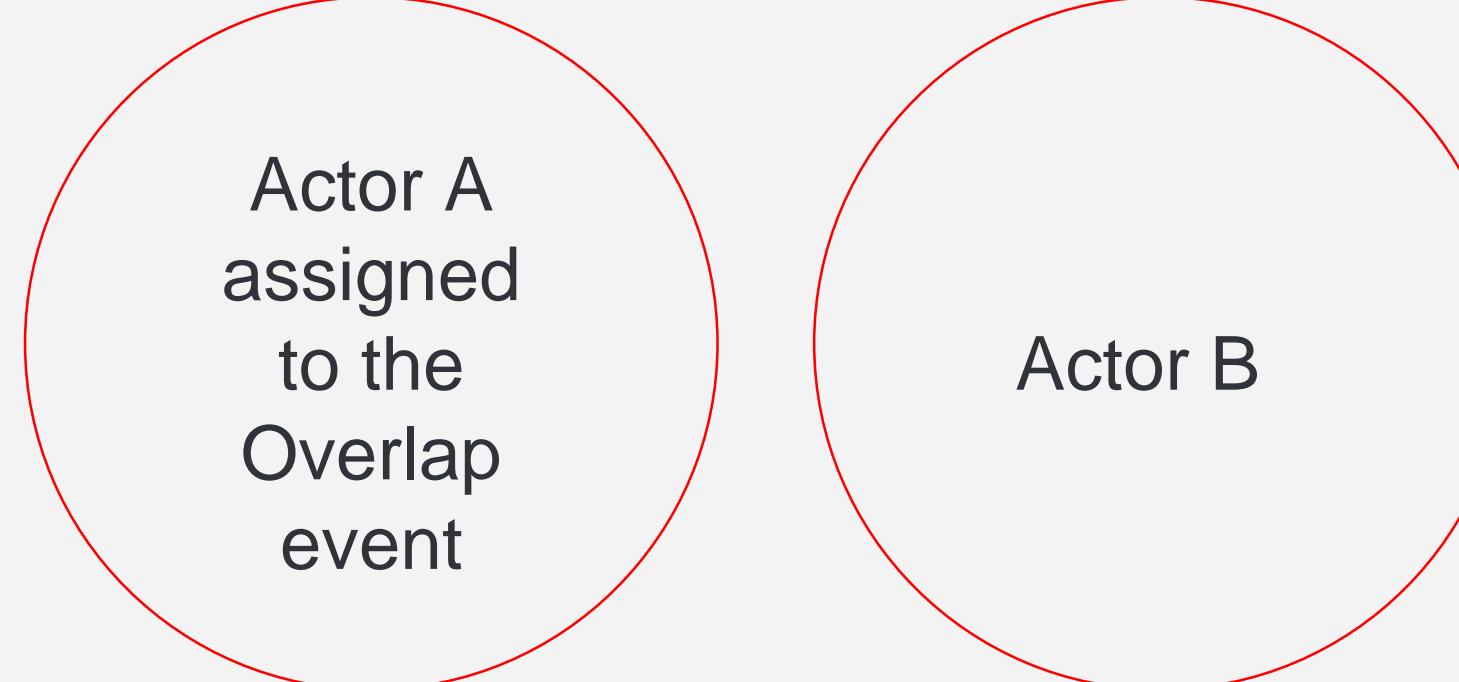
---

However, in order for Actor B to trigger the Begin Overlap event again, Actor B needs to end the overlap by moving away and then overlap Actor A a second time.

**Begin Overlap:**  
**Executes once, when collision hulls first overlap**



**End Overlap:**  
**Executes once, when collision hulls stop overlapping**



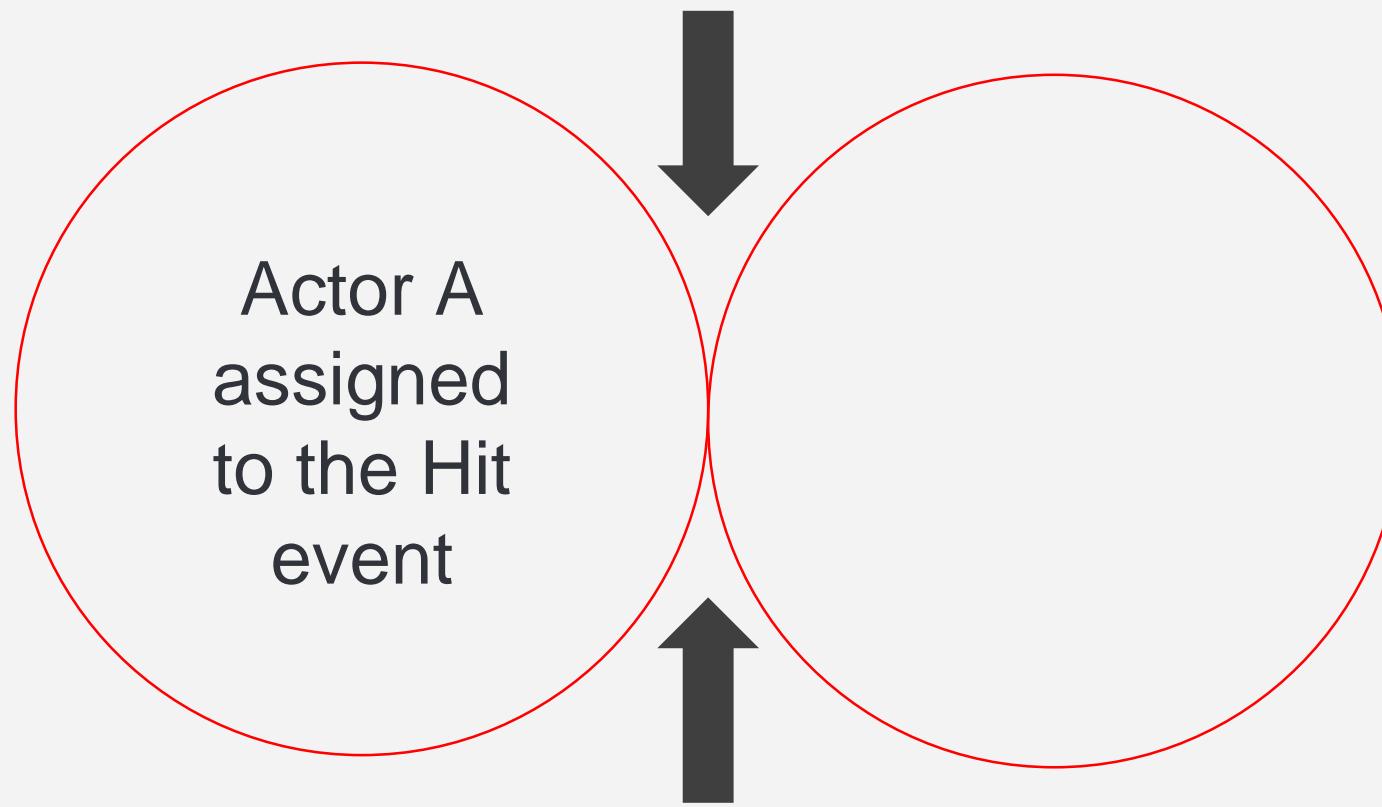


## HIT EVENTS

---

Hit events happen when a collision hull of an Actor or its components touches the collision hull of another Actor or its components.

**Hit event:**  
**Executes when collision hulls touch but do not overlap**



**Both Actors' collision hulls  
need to be set to Block for the Hit event to execute.**

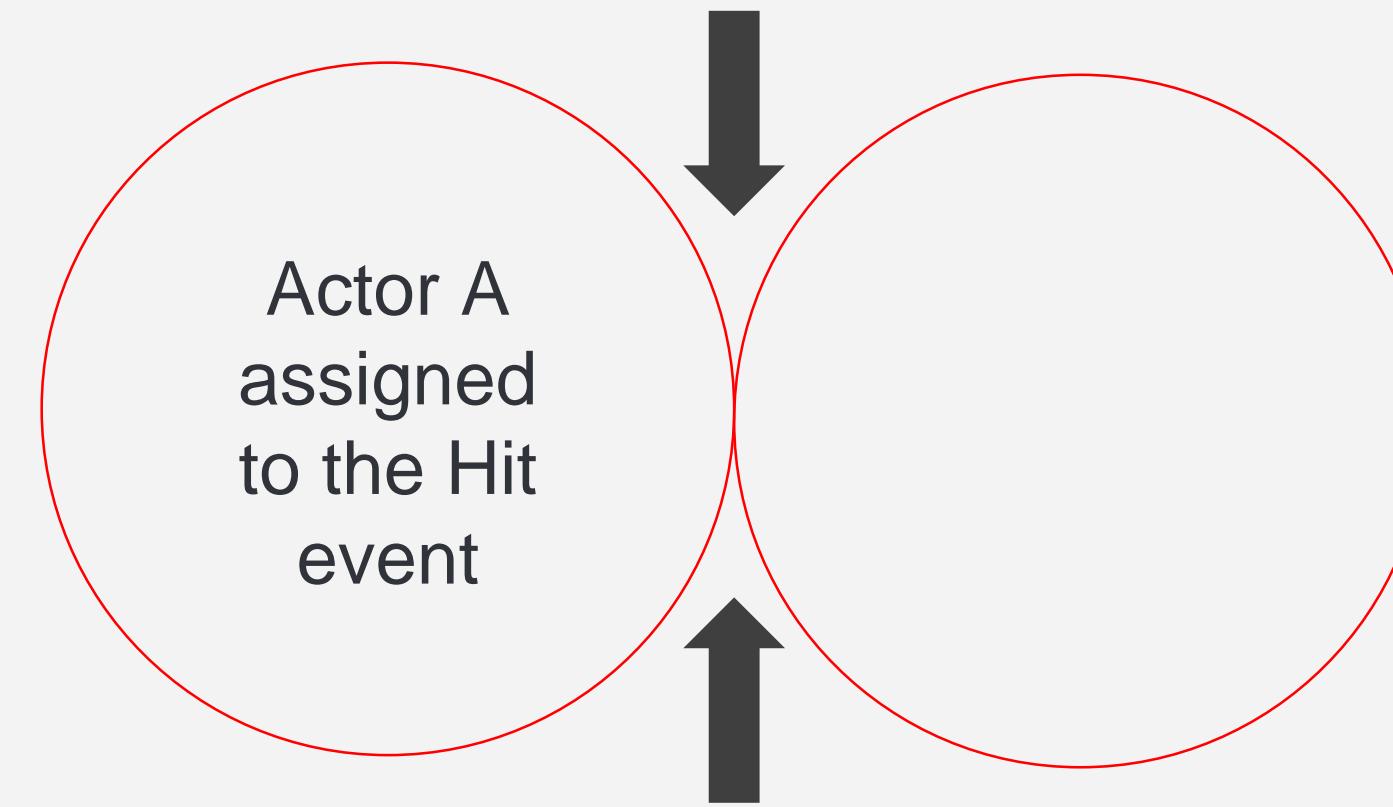


## HIT EVENTS

---

Hit events will execute multiple times as long as the Actor assigned to the event is hit by a new Actor each time or continues to hit a single Actor as it is moving.

**Hit event:**  
**Executes when collision hulls touch but do not overlap**



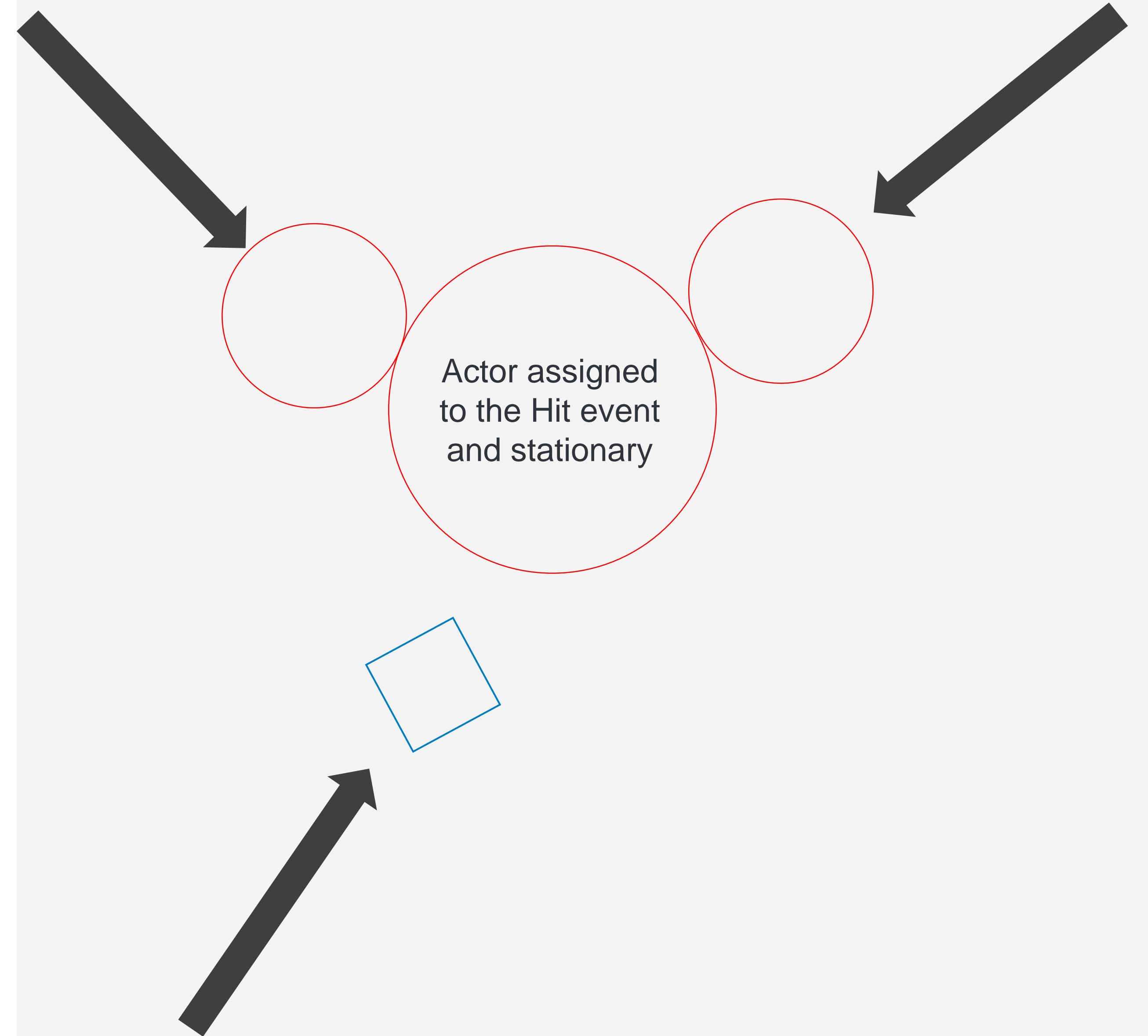
**Both Actors' collision hulls  
need to be set to Block for the Hit event to execute.**



## HIT EVENTS

---

If the Actor assigned to the Hit event is stationary and is hit by multiple other Actors, the Hit event will execute for each hit.

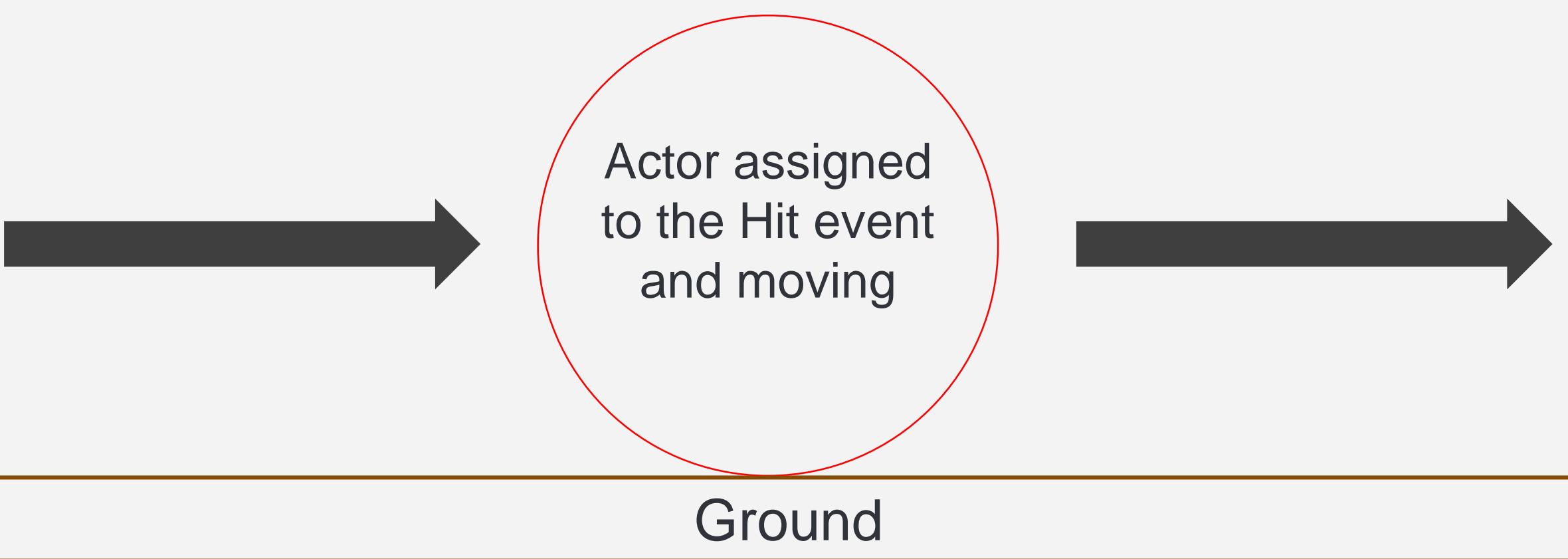




## HIT EVENTS

---

If the Actor assigned to the Hit event is moving and continues to hit the same Actor, the Hit event will also execute multiple times until the Actor comes to a rest.





## HIT EVENTS

---

Once the Actor assigned to the Hit event comes to a rest, the Hit event will stop executing unless the Actor is hit again by another Actor.

Actor assigned to the Hit event and no longer moving

Ground



# GENERATE EVENT PROPERTIES

Generate event properties are found under the Collision tab in the Level Details panel for a selected Actor. These properties allow Actors to communicate to Blueprint that a collision has occurred.

- **Generate Overlap Events** is used for Overlap events.
- **Simulation Generates Hit Events** is used for Hit events.

The screenshot shows the Unreal Engine's Level Details panel for a selected Actor named "TriggerBox". The "Collision" tab is open, displaying various collision-related properties. Two specific properties are highlighted with red boxes:

- Simulation Generates Hit Events**: This checkbox is currently unchecked (gray). It is located under the "Collision" section.
- Generate Overlap Events**: This checkbox is checked (blue), indicating it is active. It is also located under the "Collision" section.

Other visible settings include:

- Phys Material Override**: Set to "None".
- Collision Presets**: Set to "Trigger".
- Collision Enabled**: Checked.
- Object Type**: Set to "WorldDynamic".
- Collision Responses**: A table showing responses for different object types. For example, "WorldStatic" has "Ignore" checked, while "WorldDynamic" has "Overlap" checked.
- Trace Responses**: A table showing visibility and camera settings for different object types.
- Object Responses**: A table showing responses for "WorldStatic", "WorldDynamic", "Pawn", "PhysicsBody", "Vehicle", and "Destructible" objects.
- Can Character Step Up On**: Set to "Yes".



# GENERATE EVENT PROPERTIES

The Generate Overlap Events property is on by default, and Simulation Generates Hit Events is off by default.

The screenshot shows the Unreal Engine's Details panel for a component named "TriggerBox". The "Collision" section is expanded, showing the following properties:

- Simulation Generates Hit Events:** An off checkbox (unchecked) is highlighted with a red rectangle.
- Generate Overlap Events:** An on checkbox (checked) is highlighted with a red rectangle.
- Phys Material Override:** Set to "None".
- Collision Presets:** Set to "Trigger".
- Collision Enabled:** Checked.
- Object Type:** Set to "WorldDynamic".
- Collision Responses:** A table showing responses for various object types:

Object Type	Ignore	Overlap	Block
WorldStatic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
WorldDynamic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Pawn	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
PhysicsBody	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Vehicle	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Destructible	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
- Trace Responses:** A table showing visibility and camera settings for Trace Responses:

Response	Visibility	Camera
Visibility	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Camera	<input type="checkbox"/>	<input checked="" type="checkbox"/>
- Object Responses:** A table showing settings for WorldStatic, WorldDynamic, Pawn, PhysicsBody, Vehicle, and Destructible objects:

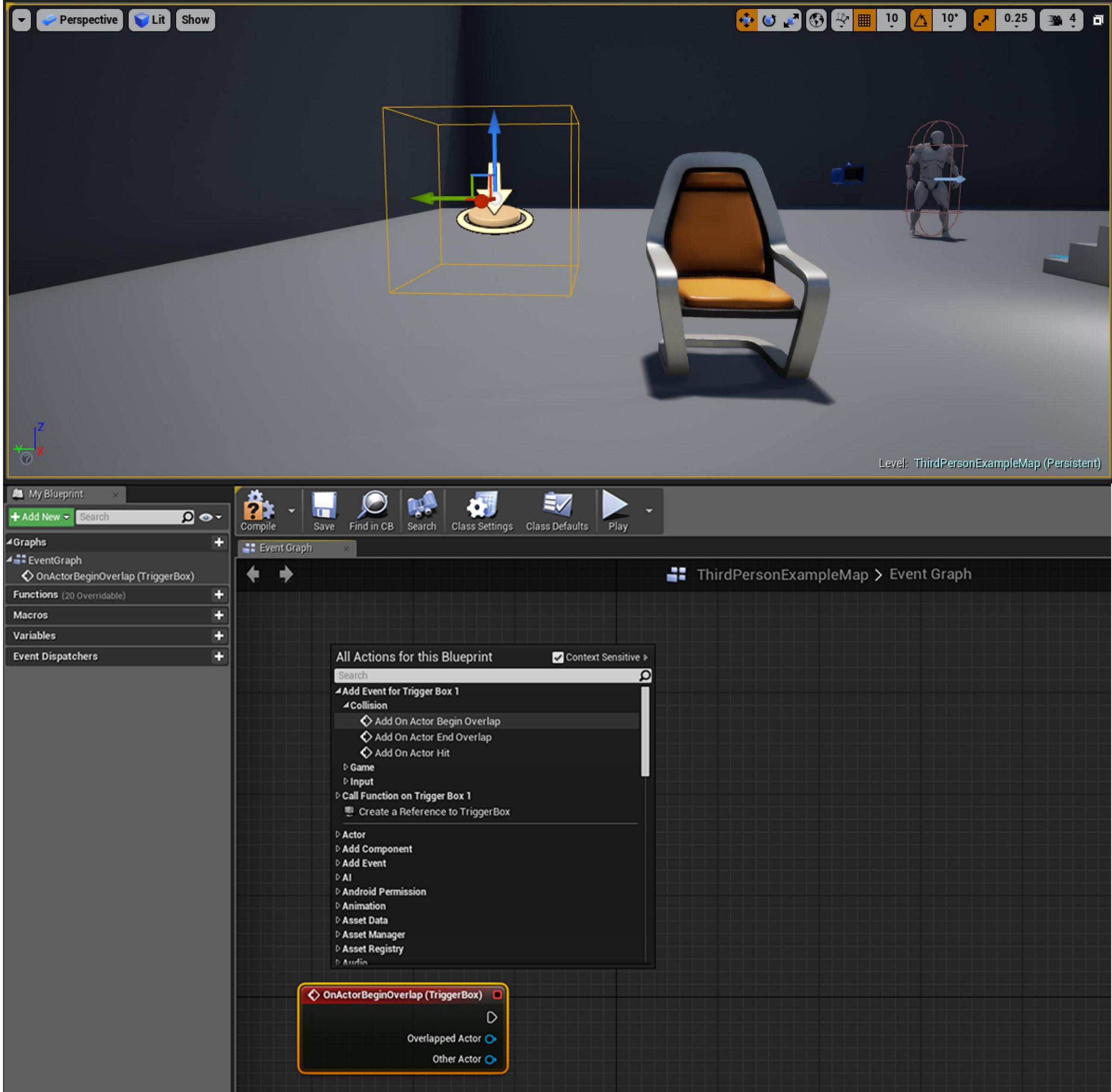
Object Type	WorldStatic	WorldDynamic	Pawn	PhysicsBody	Vehicle	Destructible
WorldStatic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WorldDynamic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Pawn	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
PhysicsBody	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Vehicle	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Destructible	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
- Can Character Step Up On:** Set to "Yes".



# ASSIGNING ACTORS TO COLLISION EVENTS

Assigning an Actor to a collision event is a simple process:

- Select the Actor in the Level and open the Level Blueprint Editor.
- Right-click in the Event Graph to bring up the Blueprint Context Menu.
- Select the collision event type you need.



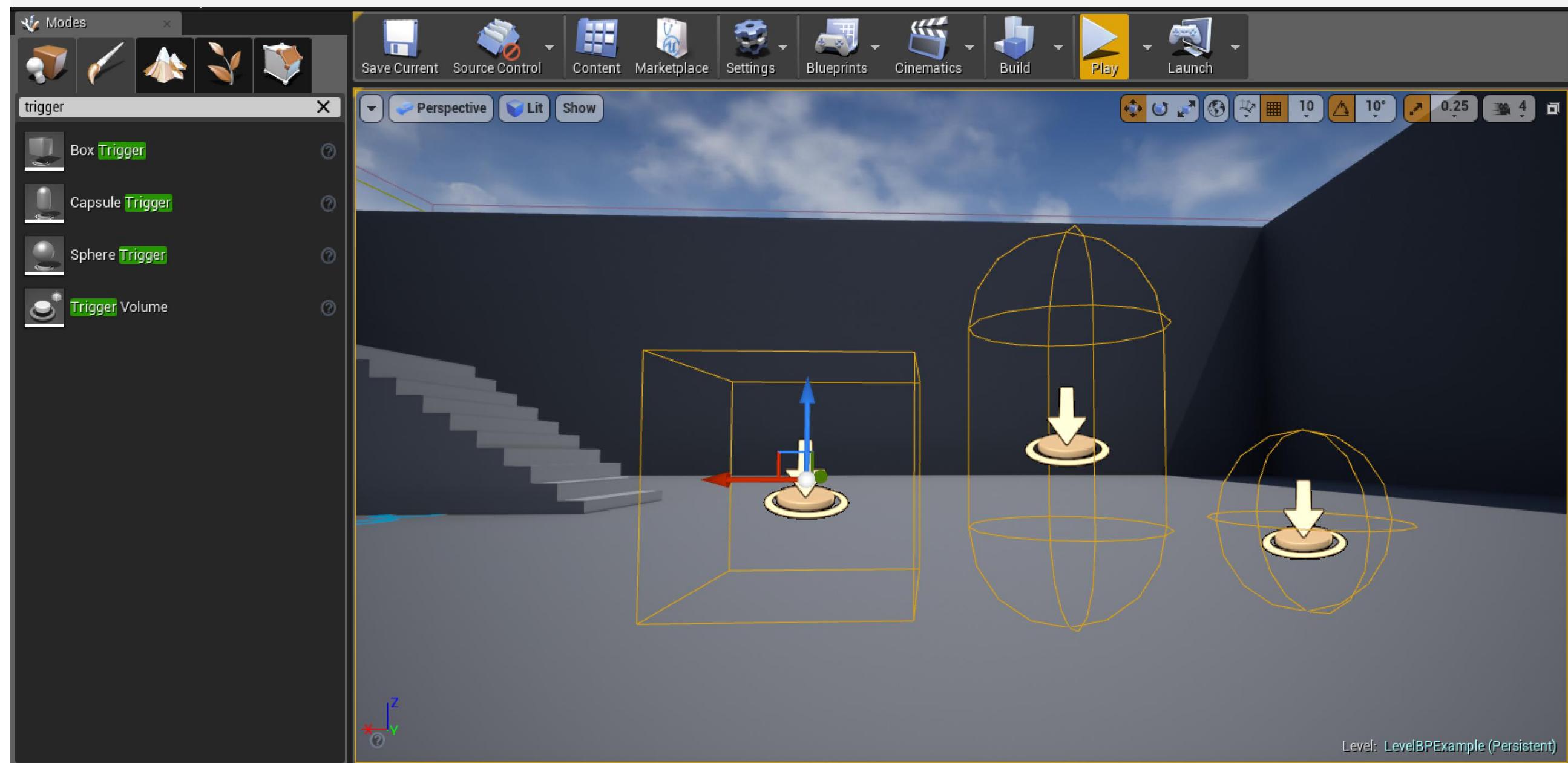


# TRIGGER ACTORS

Trigger Actors are a collection of Actors found in the Modes panel. There are three basic types:

- Box Trigger
- Capsule Trigger
- Sphere Trigger

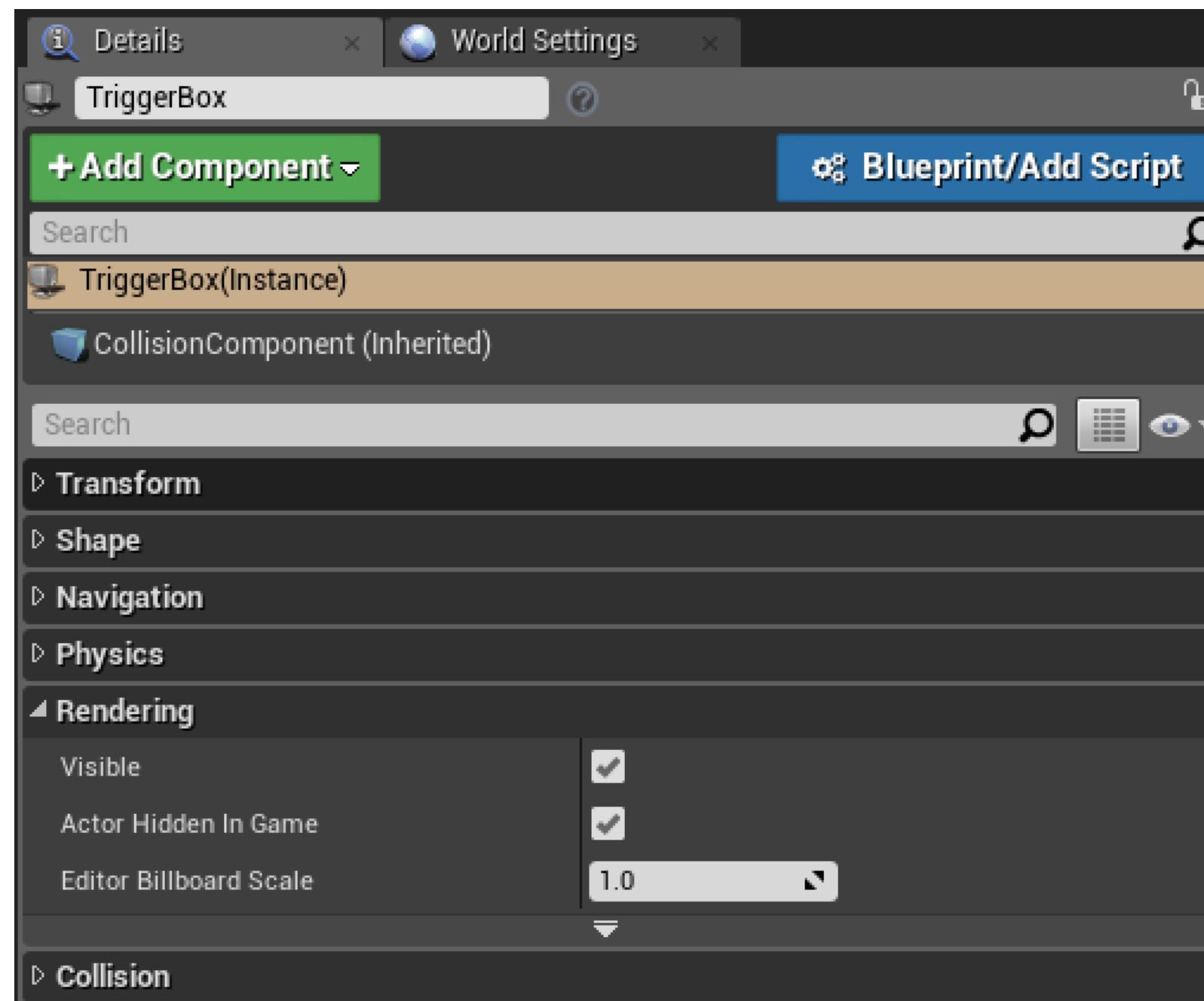
Each is used to define an area in a Level for an overlap collision event to take place. You can place as many Trigger Actors as needed in your Level and assign each to its own collision events.





# Actor Rendering Properties

When working with collision events and Trigger Actors, it helps to be able to see the Trigger Volume when playtesting your Level. By default, Actor Hidden In Game in the Rendering section of the Level Details panel is set to True. Temporarily turning this off will allow you to see the volume while playing your game. Once the collision event is working, remember to turn it back on.



# LEVEL BLUEPRINT

Collision Properties



# COLLISION PRESETS

Collision presets are a definable collection of properties that store how an individual Actor or a group of Actors should handle collisions with other Actor types in-game.

The screenshot shows the Unreal Engine Collision Preset Editor interface. The main panel displays various collision-related properties:

- Collision**:
  - Simulation Generates Hit Events: A checkbox.
  - Phys Material Override: A dropdown menu set to "None".
- Generate Overlap Events**: A checkbox.
- Collision Presets**:
  - Collision Enabled: A checkbox.
  - Object Type: A dropdown menu set to "Trigger".
    - Custom...
    - NoCollision
    - BlockAll
    - OverlapAll
    - BlockAllDynamic
    - OverlapAllDynamic
    - IgnoreOnlyPawn
    - OverlapOnlyPawn
    - Pawn
    - Spectator
    - CharacterMesh
    - PhysicsActor
    - Destructible
    - InvisibleWall
    - InvisibleWallDynamic
    - Trigger** (highlighted)
    - Ragdoll
    - Vehicle
    - UI
  - Collision Responses: A dropdown menu.
    - Trace Responses
      - Visibility
      - Camera
    - Object Responses
      - WorldStatic
      - WorldDynamic
      - Pawn
      - PhysicsBody
      - Vehicle
      - Destructible
  - Can Character Step Up On: A dropdown menu set to "Yes".
- Tags**: A section at the bottom.



# COLLISION RESPONSES

Collision presets are stored in the Collision Presets property and contain common combinations of collision responses for various Actor types, so that an Actor knows how to deal with other Actor types it may collide with. The three response settings are Ignore, Overlap, and Block.

The screenshot shows the Unreal Engine's Details panel for a 'TriggerBox' component. The 'Collision' section is expanded, and the 'Collision Responses' table is highlighted with a red rounded rectangle. The table lists various actor types and their collision responses:

Actor Type	Ignore	Overlap	Block
Trace Responses	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Visibility	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Camera	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Object Responses			
WorldStatic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
WorldDynamic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Pawn	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
PhysicsBody	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Vehicle	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Destructible	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Below the table, the 'Can Character Step Up On' dropdown is set to 'Yes'. The 'Tags' section at the bottom is currently empty.



# COLLISION RESPONSES

All collision events are tied to the Collision property settings of all the Actors involved. For example, if an Actor is assigned to an Overlap event but its collision response is set to Block and the second Actor also has a collision response set to Block, the two Actors will never overlap, so an Overlap event would never execute, but a Hit event would.

The screenshot shows the Unreal Engine's Details panel for a component named "TriggerBox". The top bar includes tabs for "Details", "World Settings", and "Blueprint/Add Script". The main area displays the "TriggerBox" component with its properties:

- TriggerBox**: A dropdown menu with "None" selected.
- + Add Component**: A button to add more components.
- Blueprint/Add Script**: A button to edit the Blueprint or add a script.
- Search**: A search bar.
- TriggerBox(Instance)**: A dropdown menu showing "CollisionComponent (Inherited)".
- Search**: Another search bar.
- Rendering**: A collapsed section.
- Collision**: An expanded section:
  - Simulation Generates Hit Events**: A checkbox that is unchecked.
  - Phys Material Override**: A dropdown menu with "None" selected.
  - Generate Overlap Events**: A checked checkbox.
  - Collision Presets**: A collapsed section.
    - Collision Enabled**: A checked checkbox.
    - Object Type**: A dropdown menu with "WorldDynamic" selected.
      - Ignore**: A checkbox that is unchecked.
      - Overlap**: A checkbox that is checked.
      - Block**: A checkbox that is checked.
  - Collision Responses**: A section with a help icon.
    - Trace Responses**:
      - Visibility**: A checked checkbox.
      - Camera**: A checked checkbox.
    - Object Responses**:
      - WorldStatic**: A checkbox that is unchecked.
      - WorldDynamic**: A checked checkbox.
      - Pawn**: A checked checkbox.
      - PhysicsBody**: A checked checkbox.
      - Vehicle**: A checked checkbox.
      - Destructible**: A checked checkbox.
  - Can Character Step Up On**: A dropdown menu with "Yes" selected.



# OBJECT TYPE

All Actors placed in a Level fall into one of six default object-type categories.

The screenshot shows the Unreal Engine's Details panel for an actor named "SM\_Chair". The "Collision" section is expanded, displaying various settings:

- Simulation Generates Hit Events: A checkbox that is unchecked.
- Phys Material Override: A dropdown menu set to "None".
- Generate Overlap Events: A checkbox that is unchecked.
- Collision Presets: A dropdown menu set to "Custom...".
- Collision Enabled: A dropdown menu set to "Collision Enabled (Query and Physics)".
- Object Type: A dropdown menu set to "WorldStatic", with other options like "WorldDynamic", "Pawn", "PhysicsBody", "Vehicle", and "Destructible" listed below it.
- Collision Responses: A section with checkboxes for Trace Responses (Visibility, Camera), Object Responses (WorldStatic, WorldDynamic, Pawn, PhysicsBody, Vehicle, Destructible), and Can Character Step Up On (Yes).
- Can Character Step Up On: A dropdown menu set to "Yes".



# OBJECT TYPE

The Object Type property declares what category an Actor falls into so that when it collides with another Actor, that Actor will know how to respond to it.

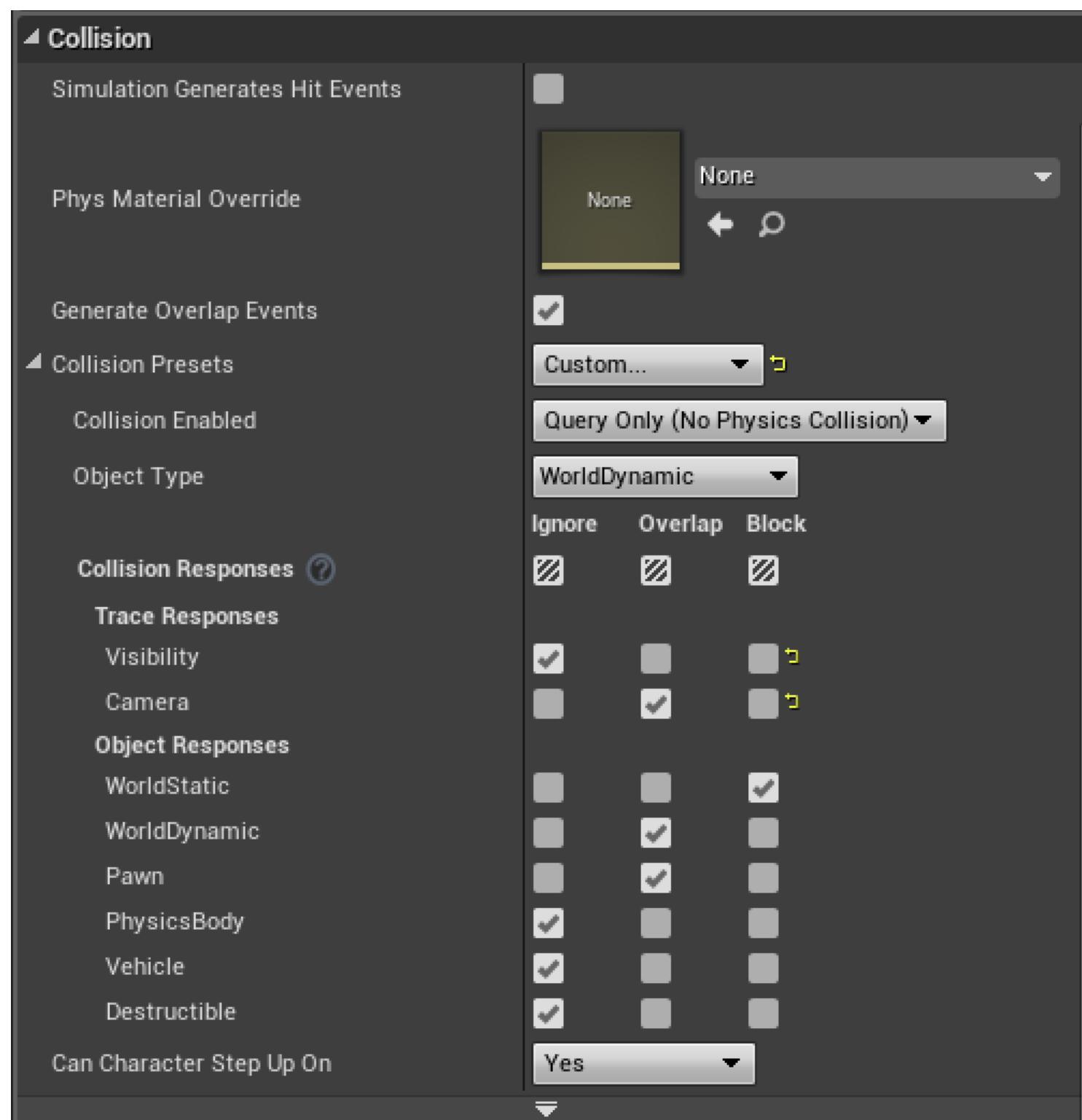
The screenshot shows the Unreal Engine's World Settings panel for a component named "TriggerBox". The "Collision" section is expanded, and the "Collision Responses" section is highlighted with a red oval. This section contains a table where rows represent different actor types and columns represent collision responses: Ignore, Overlap, and Block. The "Trigger" row is selected, indicated by a blue highlight. The "WorldDynamic" row is also highlighted with a blue box.

	Ignore	Overlap	Block
Trigger	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WorldDynamic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
WorldStatic	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Pawn	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
PhysicsBody	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Vehicle	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Destructible	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>



# Custom Collision Presets for an Individual Actor

You can customize an individual Actor's collision responses simply by picking Custom Preset. This will allow you to manually edit how the Actor should respond to other Actor types. It will also allow you to change the Actor's object type.





# Setting Collision Presets for an Entire Project

You can define new object types and collision presets for an entire project. Doing so will speed up production if your project has a lot of specific needs when dealing with collision responses.

On the Level Editor menu bar, choose Edit > Project Settings > Collision.

The screenshot shows the 'Project Settings' window in the Unreal Engine Level Editor. The left sidebar lists categories: GameplayTags, Maps & Modes, Movies, Packaging, Supported Platforms, Target Hardware, Game (Asset Manager), and Engine (AI System, Animation, Audio, Collision, Console, Cooker, Crowd Manager, End-User Settings, Gameplay Debugger, Garbage Collection, General Settings, Input, Navigation Mesh, Navigation System, Network, Niagara, Physics, Rendering, Rendering Overrides (Local)). The main content area is titled 'Engine - Collision' with the sub-section 'Collision'. It displays a table of collision profiles:

Name	Collision	Object Type	Description
NoCollision	No Collision	WorldStatic	No collision
BlockAll	Collision Enabled (Query and IWorldStatic)	WorldStatic	WorldStatic object that blocks all actors by default. All new
OverlapAll	Query Only (No Physics CollidesWithWorldStatic)	WorldStatic	WorldStatic object that overlaps all actors by default. All ne
BlockAllDynamic	Collision Enabled (Query and IWorldDynamic)	WorldDynamic	WorldDynamic object that blocks all actors by default. All n
OverlapAllDynamic	Query Only (No Physics CollidesWithWorldDynamic)	WorldDynamic	WorldDynamic object that overlaps all actors by default. All
IgnoreOnlyPawn	Query Only (No Physics CollidesWithWorldDynamic)	WorldDynamic	WorldDynamic object that ignores Pawn and Vehicle. All ot
OverlapOnlyPawn	Query Only (No Physics CollidesWithWorldDynamic)	WorldDynamic	WorldDynamic object that overlaps Pawn, Camera, and Vehi
Pawn	Collision Enabled (Query and IPawn)	Pawn	Pawn object. Can be used for capsule of any playable cha
Spectator	Query Only (No Physics CollidesWithPawn)	Pawn	Pawn object that ignores all other actors except WorldStat
CharacterMesh	Query Only (No Physics CollidesWithPawn)	Pawn	Pawn object that is used for Character Mesh. All other char
PhysicsActor	Collision Enabled (Query and IPhysicsBody)	Simulating actors	
Destructible	Collision Enabled (Query and IDestructible)	Destructible actors	
InvisibleWall	Collision Enabled (Query and IWorldStatic)	WorldStatic	WorldStatic object that is invisible.
InvisibleWallDynamic	Collision Enabled (Query and IWorldDynamic)	WorldDynamic	WorldDynamic object that is invisible.
Trigger	Query Only (No Physics CollidesWithWorldDynamic)	WorldDynamic	WorldDynamic object that is used for trigger. All other chan
Ragdoll	Collision Enabled (Query and IPhysicsBody)	Simulating Skeletal Mesh Component. All other channels w	
Vehicle	Collision Enabled (Query and IVehicle)	Vehicle	Vehicle object that blocks Vehicle, WorldStatic, and WorldD
UI	Query Only (No Physics CollidesWithWorldDynamic)	WorldStatic	WorldStatic object that overlaps all actors by default. All ne

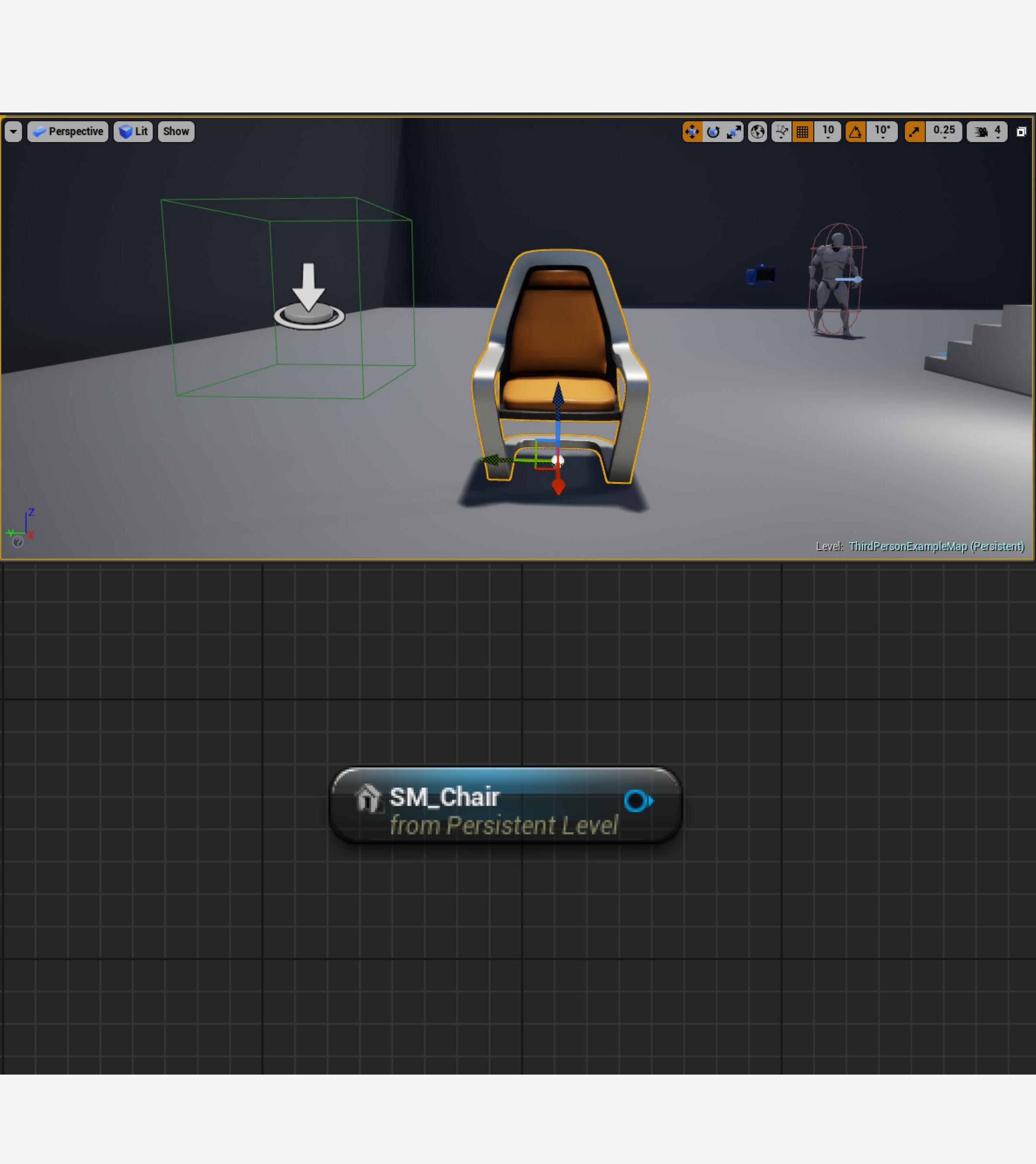
# LEVEL BLUEPRINT

Assigning Actors to Reference Variables



# ACTOR REFERENCE VARIABLES

An Actor reference variable represents an Actor and all of its properties in Blueprint. It is how you refer to an Actor whose properties you want to modify in Blueprint.



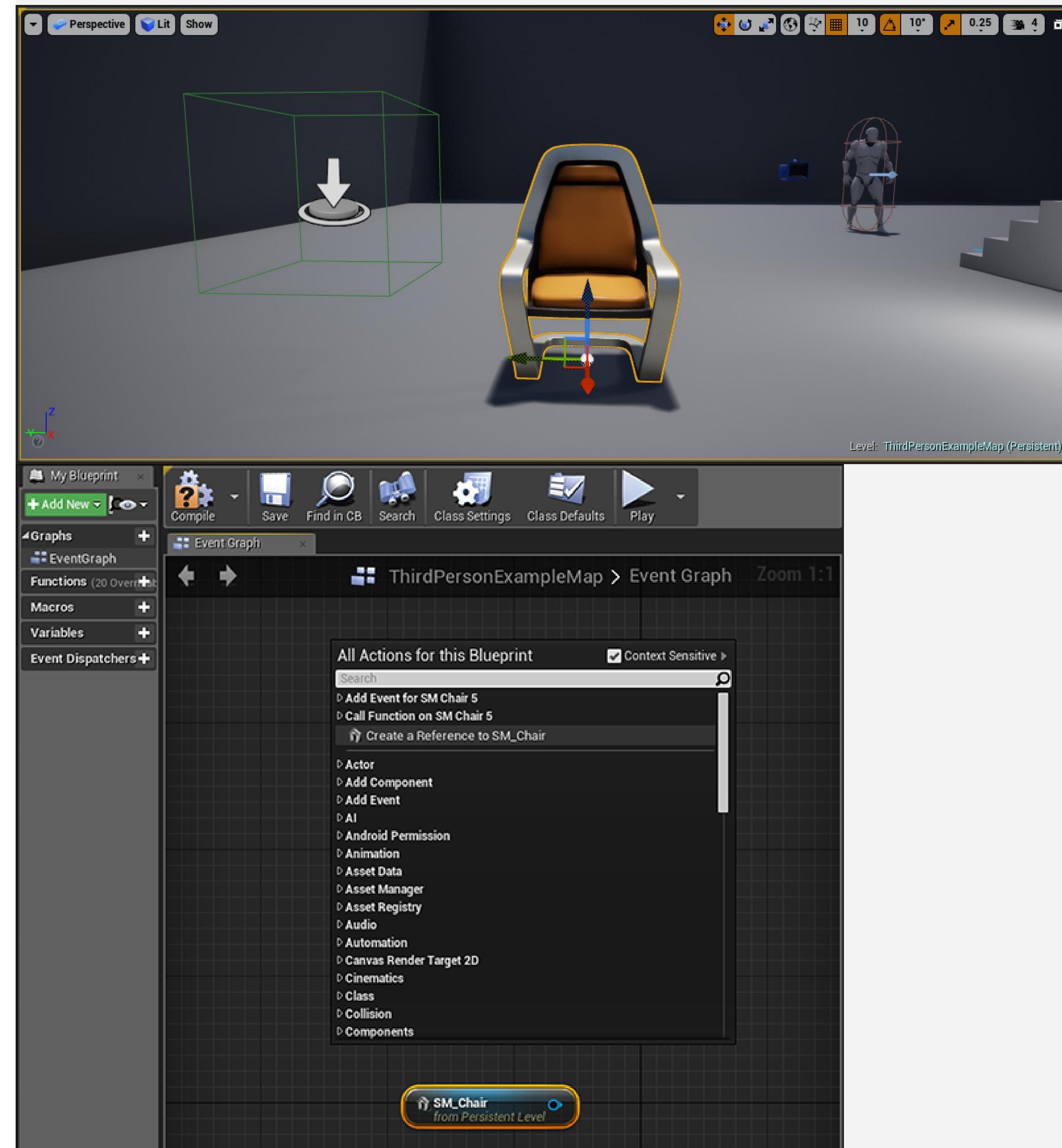


# ACTOR REFERENCE VARIABLES

You can only assign Actors already placed in the Level to an Actor reference variable in the Level Blueprint.

To assign an Actor to an Actor reference variable:

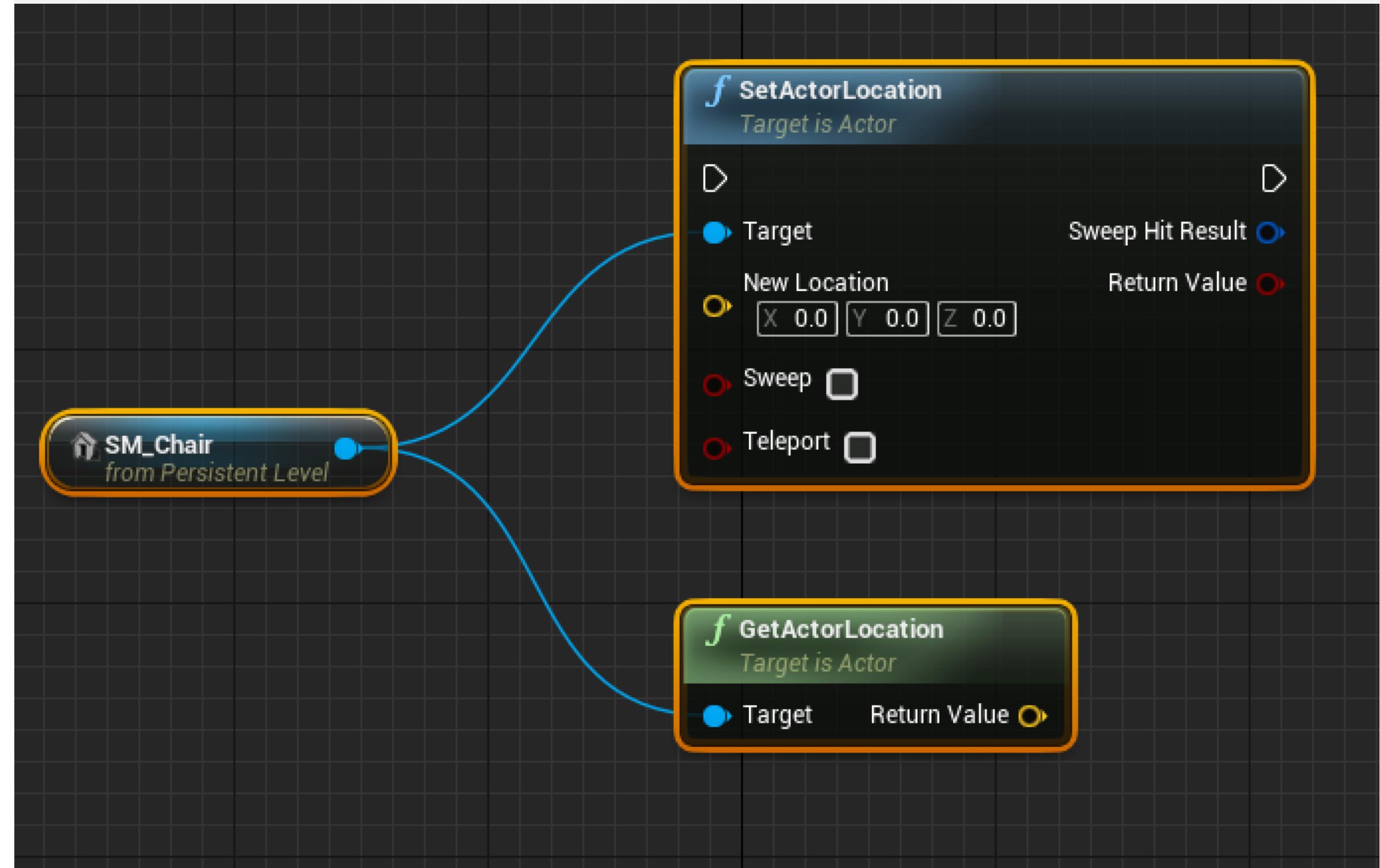
- Select the Actor in the Level and open the Level Blueprint Editor.
- Right-click in the Event Graph to bring up the Blueprint Context Menu.
- Select Create a Reference to [Actor] to add a variable for the selected Actor.





# CHANGING AN ACTOR'S PROPERTIES

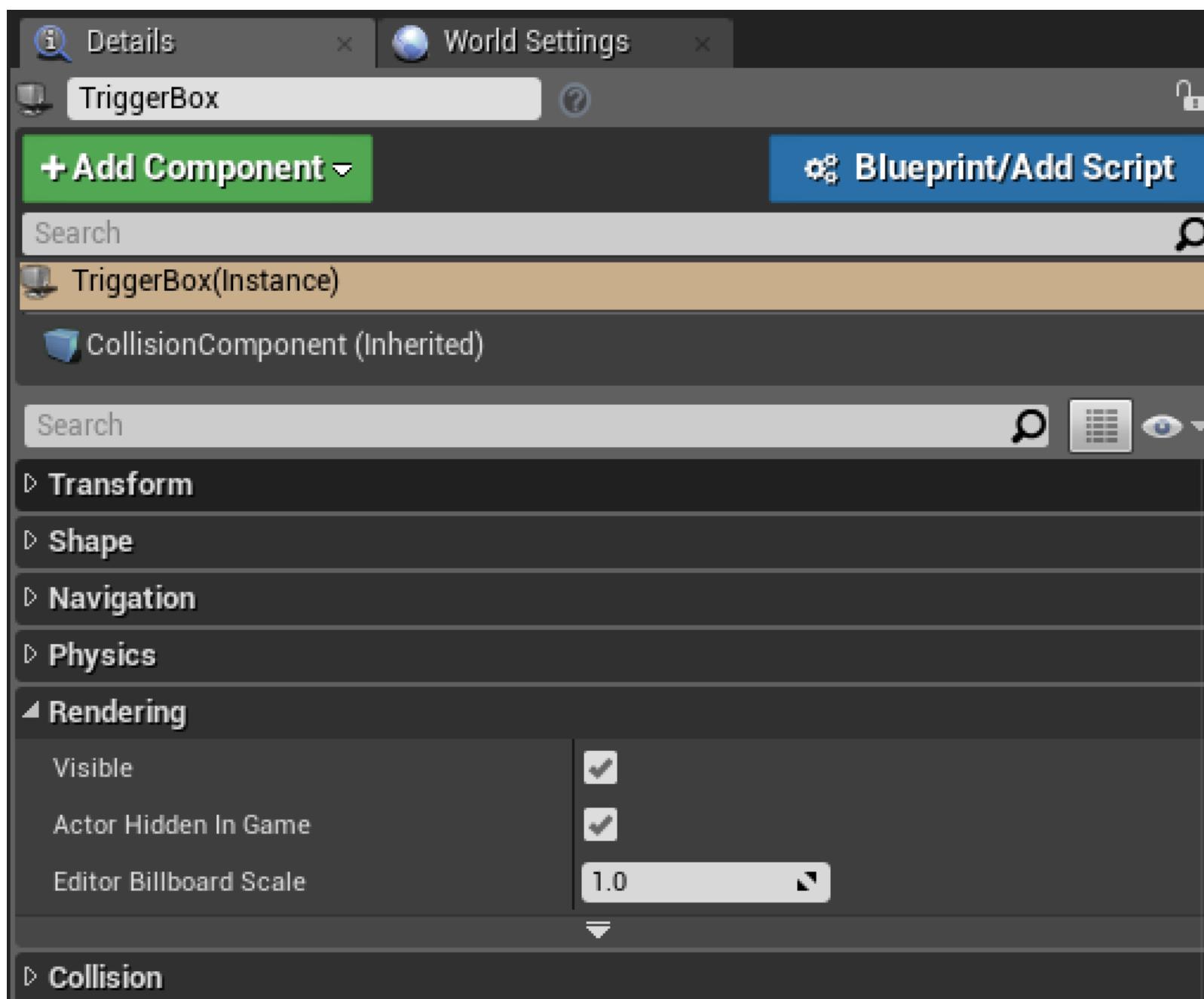
Once an Actor has been assigned to a reference variable, you can get or set the Actor's properties. For example, you can use a function to change its location in the Level.





# Actor Properties

If you are not sure what kind of properties an Actor has, select the Actor in the Level and look at its properties in the Level Details panel.





# ACTOR COMPONENTS

All Actors placed in a Level have common properties on the Actor level, such as transform and rendering settings.

The screenshot shows the Unreal Engine's Details panel for an actor named "TriggerBox". The top bar includes tabs for "Details" and "World Settings". Below the tabs, there is a search bar and a button labeled "+ Add Component". A blue button on the right says "Blueprint/Add Script". The main area displays a list of components:

- TriggerBox (Instance)
- CollisionComponent (Inherited)

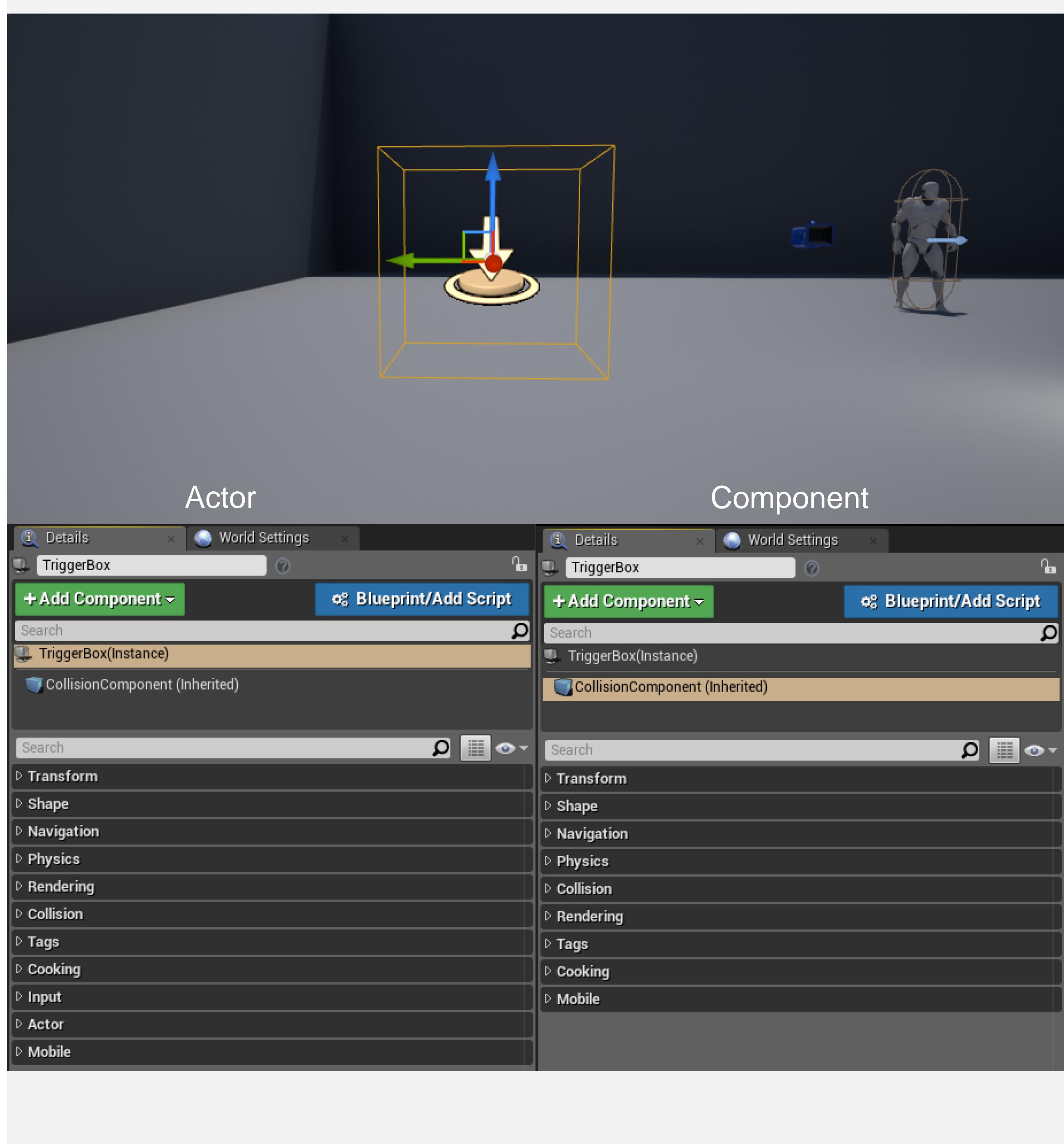
Below this, a sidebar lists various component categories with expandable arrows:

- Transform
- Shape
- Navigation
- Physics
- Rendering
- Collision
- Tags
- Cooking
- Input
- Actor
- Mobile



# COMPONENT PROPERTIES

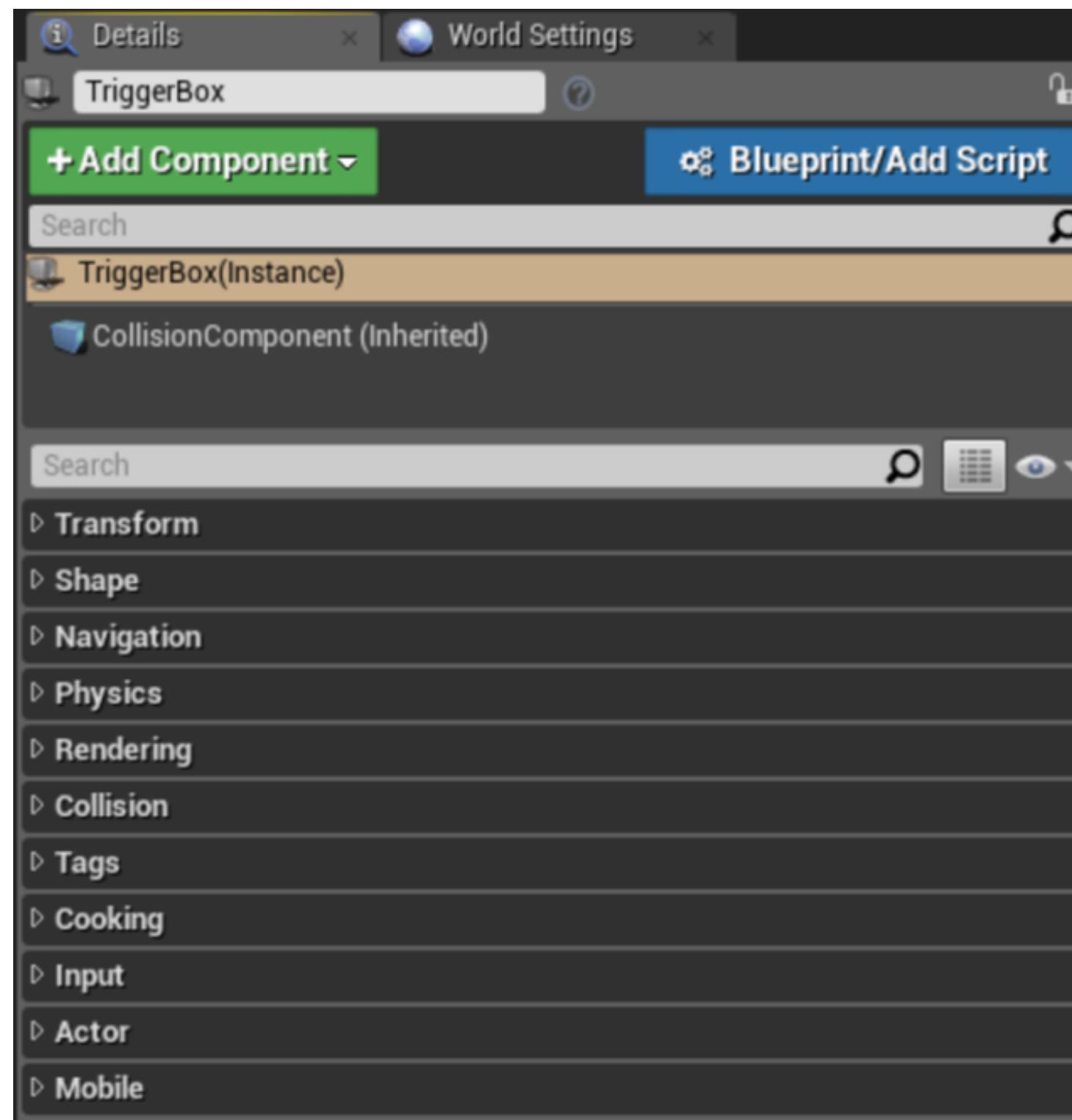
Other properties of an Actor are on the component level. A component is a subobject element of an Actor. For example, a Box Trigger Actor has a Collision component.





# Actors and Components

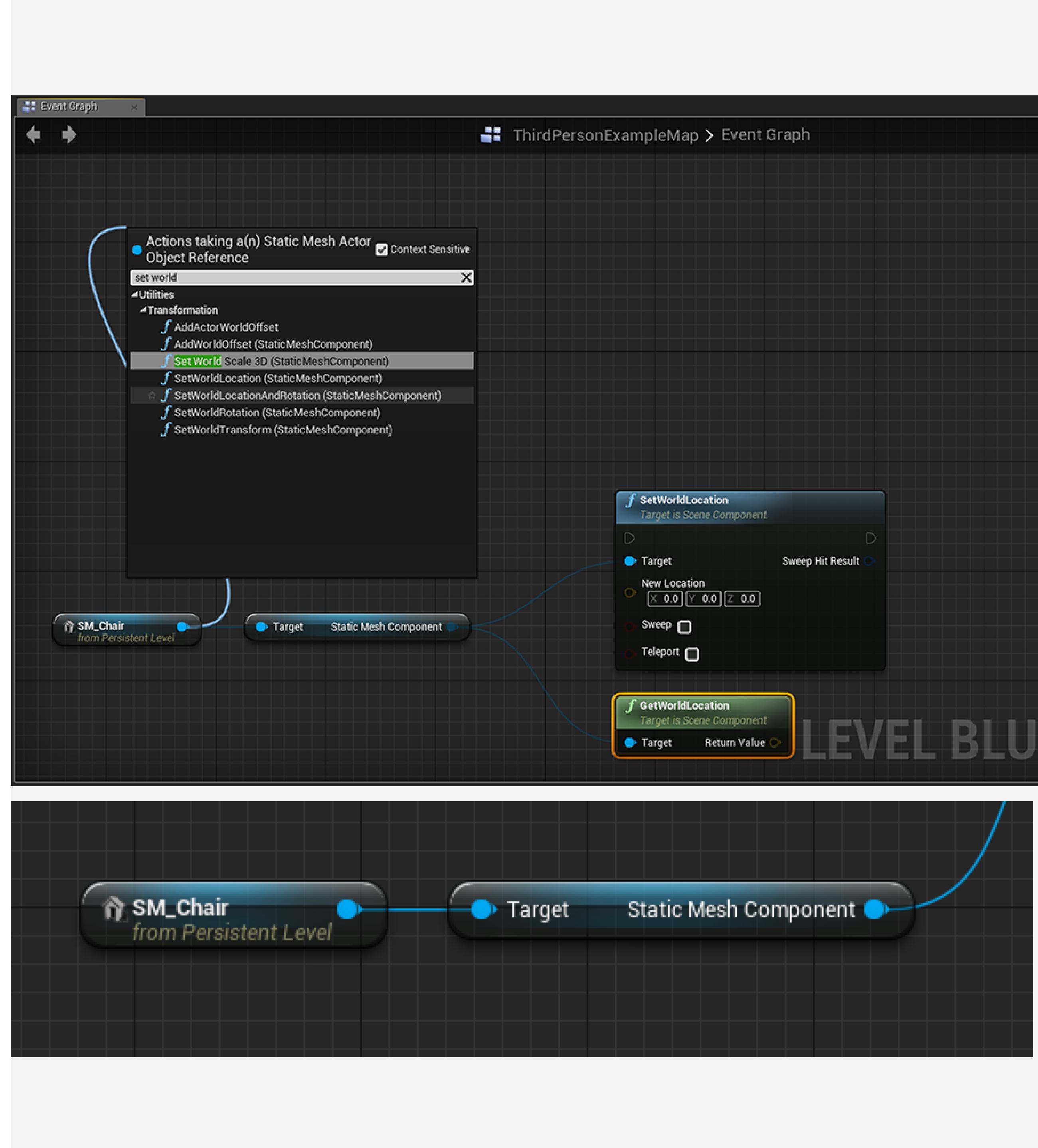
Many of the default classes already provided in the Editor show the component properties on the Actor level. For example, the Actor properties for the Box Trigger shows the Shape property tab on the Actor level, but the shape property is actually a property of the Collision component.





# CHANGING AN ACTOR'S PROPERTIES

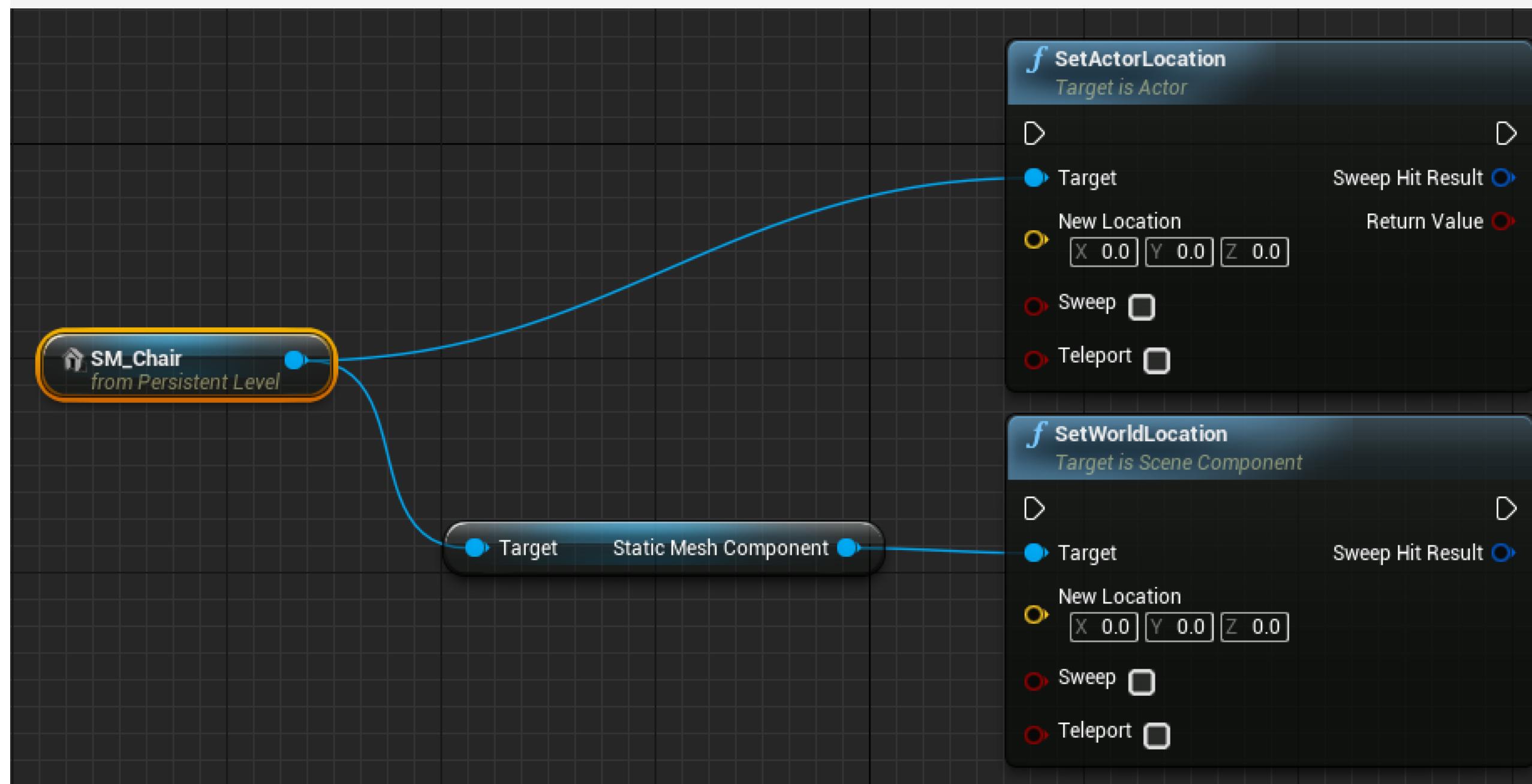
You can use a component reference to get or set a property on a component of an Actor.





# CHANGING AN ACTOR'S PROPERTIES

Setting the properties of an Actor or its components requires a function that targets the Actor or the Actor's components. In the example on the right, both functions change the location of their target, but the top function targets the Actor and the bottom one targets a component of the Actor.



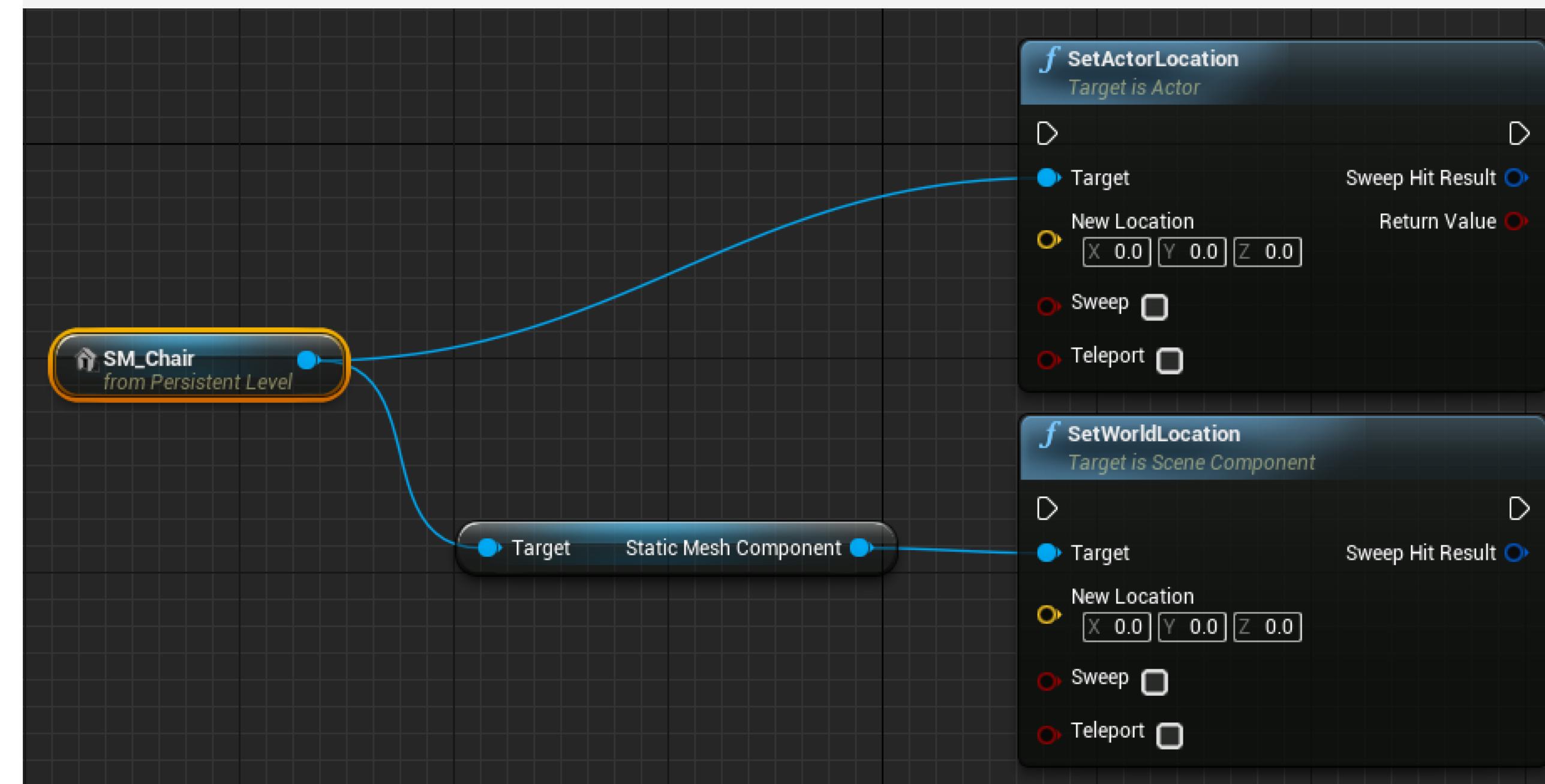
# LEVEL BLUEPRINT

Walkthrough



# CREATE A NEW PROJECT

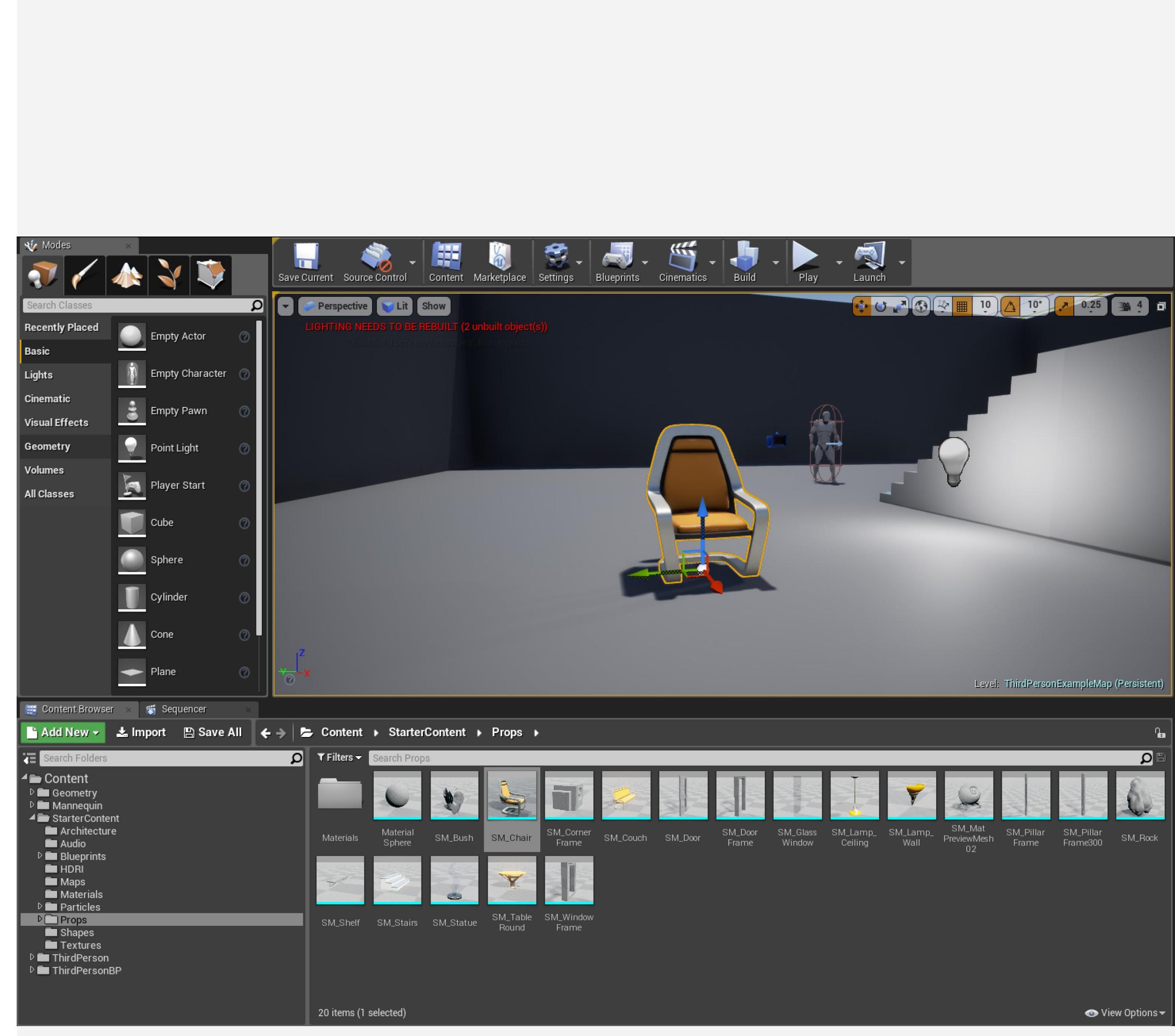
Create a new project using the Third Person template with starter content.





# ADD ACTORS TO WORK WITH

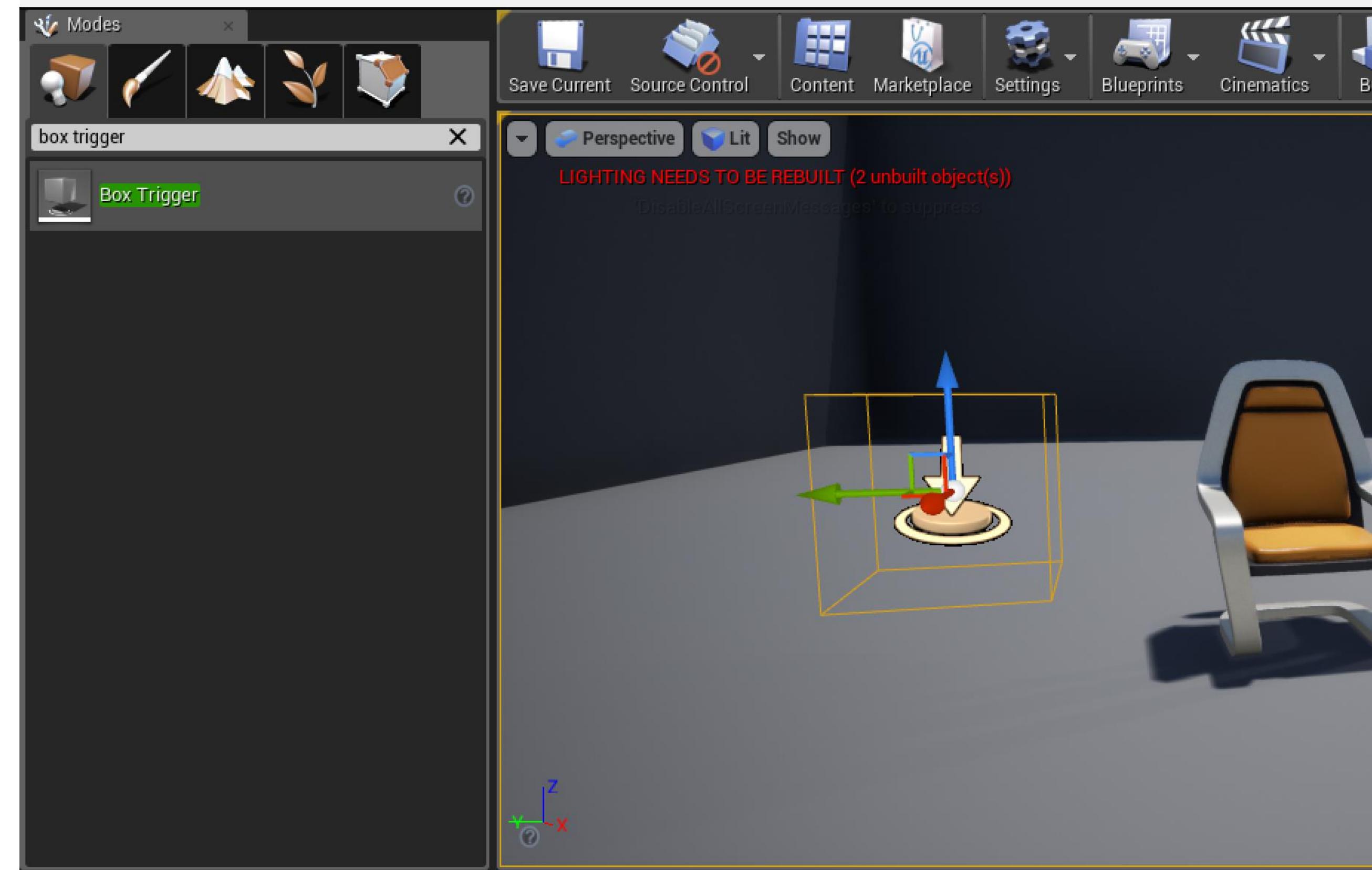
Add the SM\_Chair Static Mesh asset from the Content Browser to the default Level.





## ADD ACTORS TO WORK WITH

Now add a Box Trigger Actor. In the Modes panel search box, type “Box Trigger” to find the Box Trigger and drag it into your Level.





# CHANGE THE BOX TRIGGER'S RENDERING SETTING

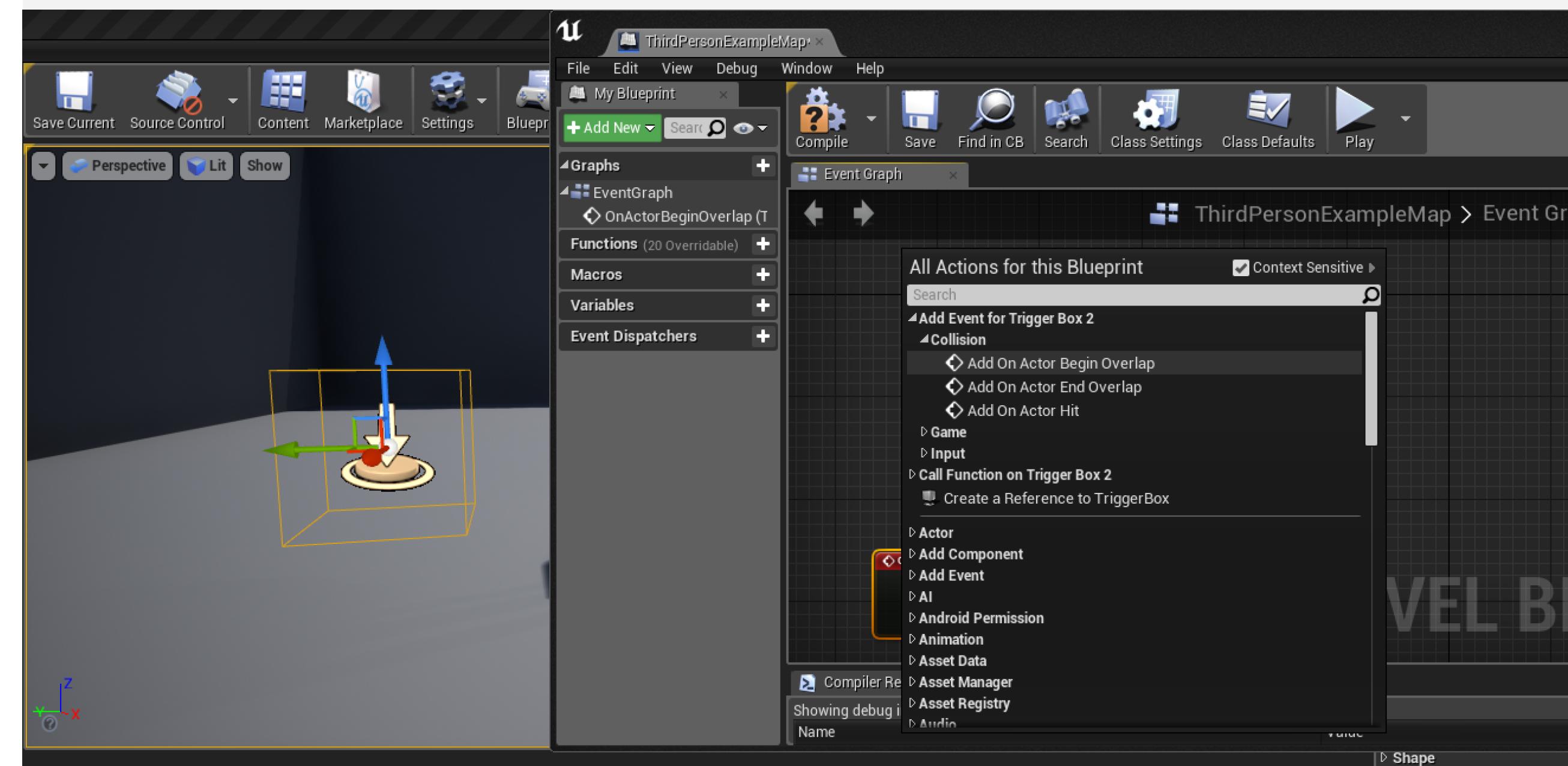
Temporarily change the Box Trigger Actor's rendering setting so you can see it in-game. Select the Box Trigger, and in the Level Details panel under Rendering, turn off Actor Hidden In Game.





## ASSIGN THE BOX TRIGGER TO AN OVERLAP EVENT

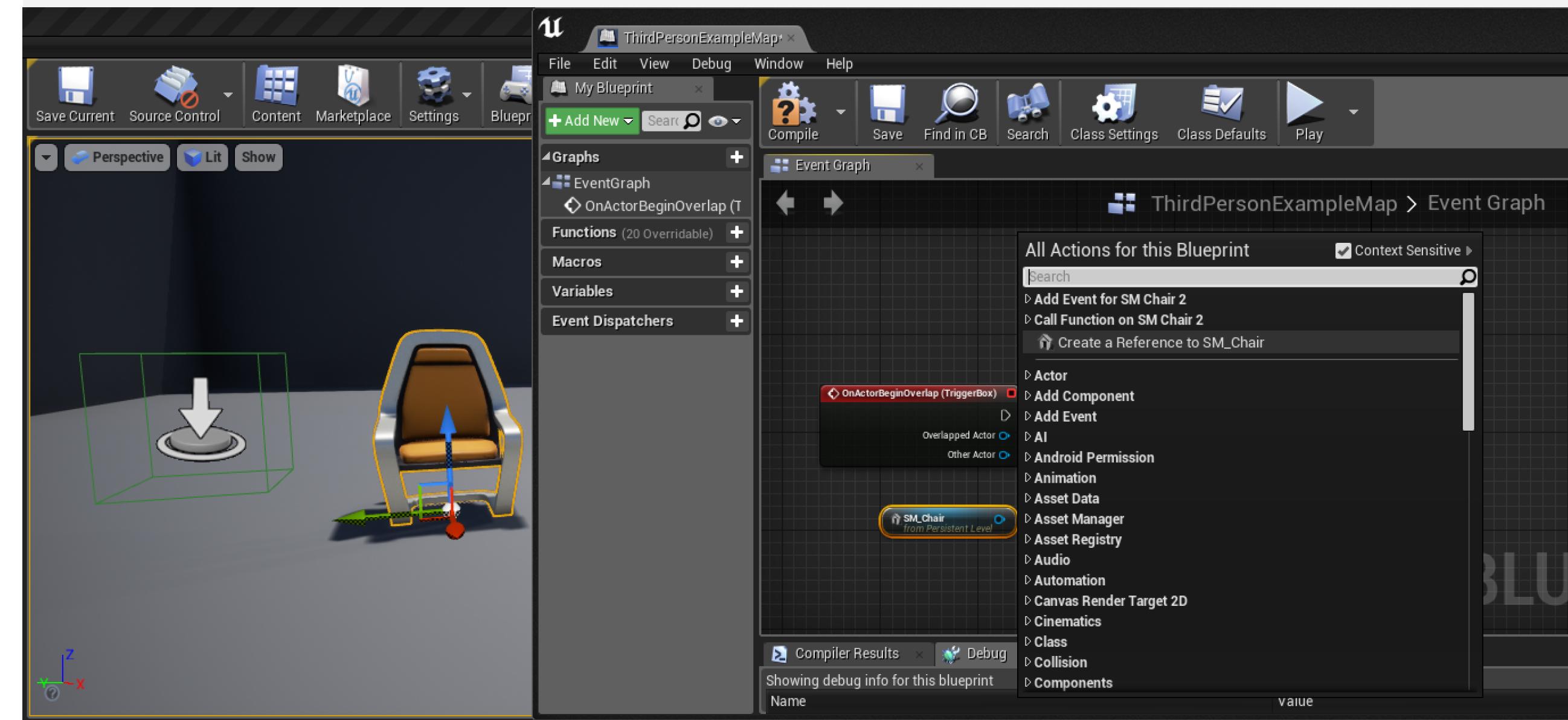
Select the Box Trigger Actor in the Level and open up the Level Blueprint Editor. Right-click in the Event Graph to bring up the Blueprint Context Menu and select Add On Actor Begin Overlap to assign the Box Trigger Actor to an overlap collision event.





# CREATE A REFERENCE TO THE SM\_CHAIR ACTOR

Select the SM\_Chair Static Mesh Actor in the Level. In the Level Blueprint, right-click in the Event Graph to bring up the Blueprint Context Menu and select Create a Reference to SM\_Chair to assign the Actor to a reference variable.





# CHANGE THE MATERIAL OF THE SM\_CHAIR ACTOR

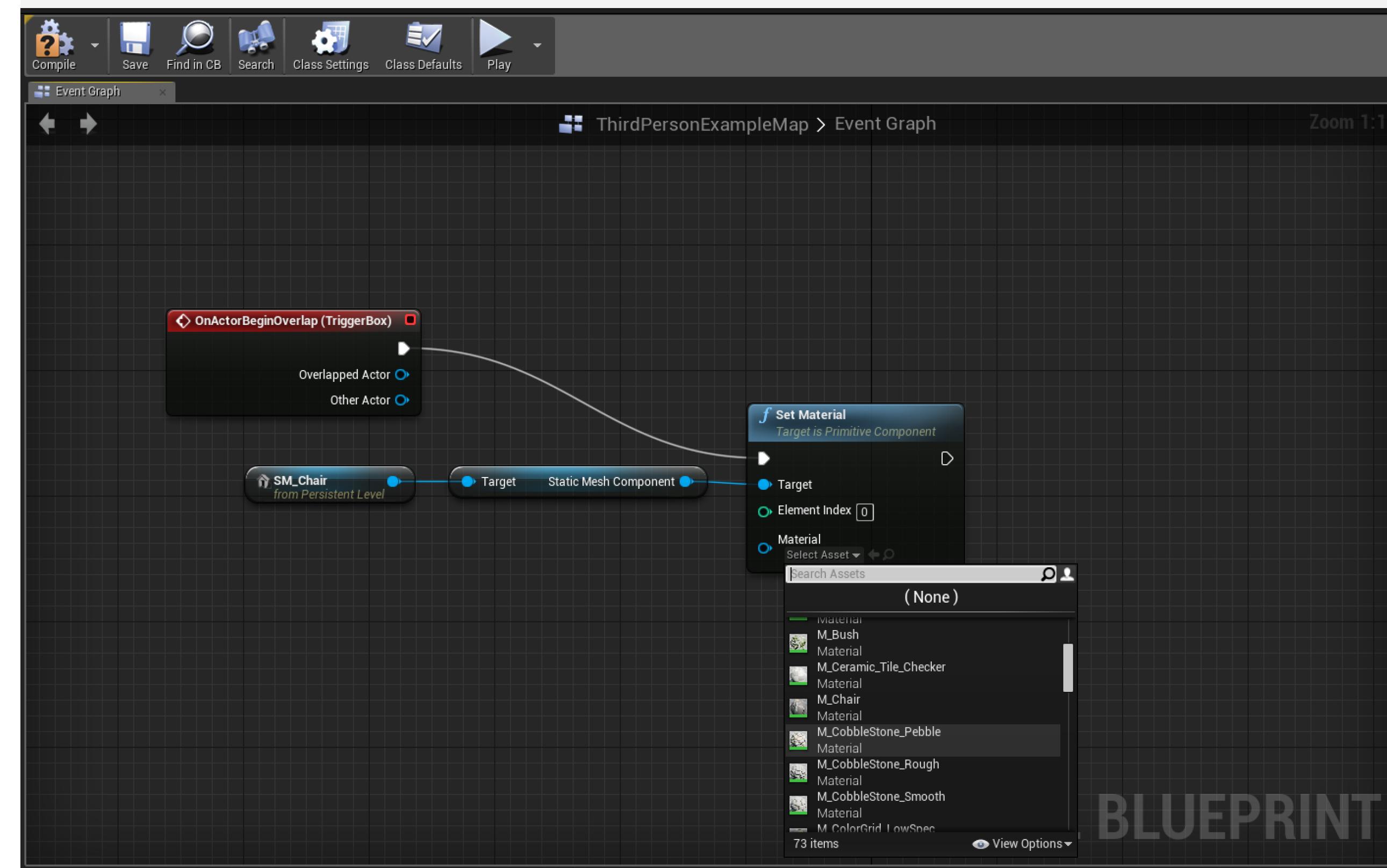
In the Level Blueprint, drag a wire from the blue data out pin on the Chair Actor reference variable and release it to bring up the Blueprint Context Menu. In the search box, type “set Material” and select Set Material (StaticMeshComponent) to add a Set Material function that targets the mesh component of the Actor.

The screenshot shows the Unreal Engine Level Blueprint Editor interface. The main area displays the Event Graph for the 'OnActorBeginOverlap' event. A trigger box actor is overlaid on a chair actor. The graph consists of two nodes: 'OnActorBeginOverlap (TriggerBox)' and 'SM\_Chair from Persistent Level'. A wire connects the 'Overlapped Actor' pin of the first node to the 'Target' pin of the second node. A third node, 'Set Material', is also present in the graph. The right side of the screen features the Blueprint Context Menu, which is open and set to the 'Material' category. The search bar in the menu contains the text 'set material'. Several options are listed under the 'Material' section, including 'Set Material (StaticMeshComponent)', 'Set Material by Name (StaticMeshComponent)', 'Set Scalar Parameter Value on Materials (StaticMeshComponent)', and 'Set Vector Parameter Value on Materials (StaticMeshComponent)'. The bottom right corner of the screen has the text 'LEVEL BLUEPRINT'.



# CHANGE THE MATERIAL OF THE SM\_CHAIR ACTOR

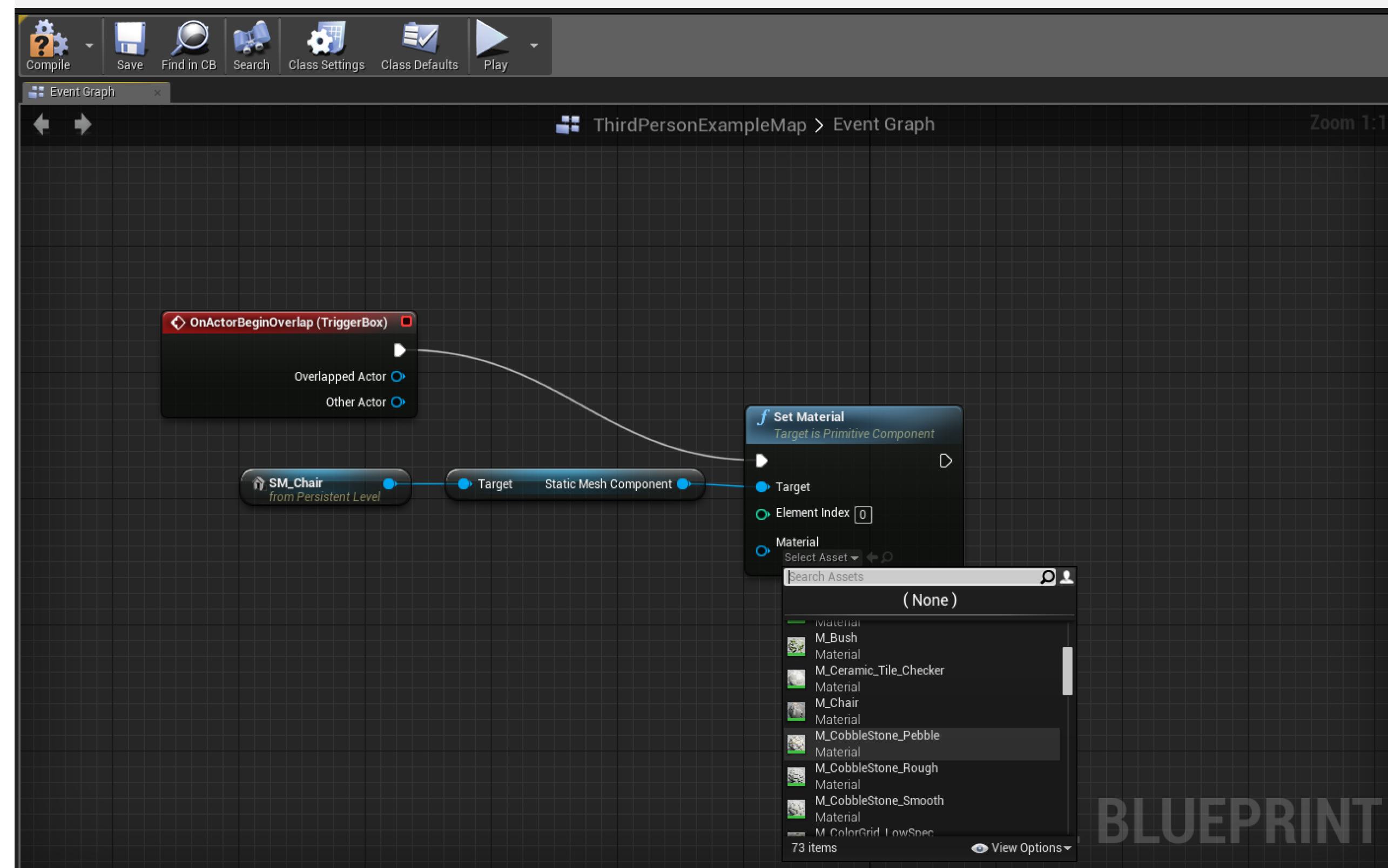
Drag a wire from the Overlap event's exec out pin to the Set Material function's exec in pin to link them together. Then click on the Material property on Set Material and select a new Material from the drop-down list.





# CHANGE THE MATERIAL OF THE SM\_CHAIR ACTOR

Compile the Blueprint and preview the Level. Move the Character Pawn over the Box Trigger Actor to see the Material on the SM\_Chair Actor change.





## END OVERLAP EVENT

Now repeat the process, but this time assign the Box Trigger to an End Overlap event and change the Material again.

