Posting Sounds on Collision

> This tutorial will show how to post sounds upon a collider experiencing collision, both for Unity and Unreal (5.1).

UNREAL (5.1)

Collision can happen in many different ways: The harder you kick a soccer ball, the louder the impact should sound, and vice versa. Using Unreal's Physics System and Wwise RTPCs, we can create a collision detection system that will only post a sound event on collision if the impact reaches a certain threshold.

Linked here is a web tutorial on Audio Kinetic's website for a reiterated breakdown on the process.

a.PNG

In Wwise, you need to create an RTPC in the Game Syncs tab of your Project Explorer window. I named mine *Collision_Velocity* and upon making it, went to the sound SFX object in my audio tab that I am using for our collision scenario: *Cube_Collision*.

By clicking this SFX object and navigating to RTPC tab in the Property Editor, I can add effects onto a Y Axis and on the X axis, define the RTPC these effects should be tied to. I used two effects, Lo-pass and Volume, and tied both of these to my *Collision_Velocity* RTPC.

> *If you right click the line curves, you can change the type of curve an effect undergoes!*

b.PNG

When your sound if configured how you want it and the RTPC is created and connected to the SFX object, you need to create a Wwise Event to drop your sound into, enabling us to post this event in Unreal upon regenerating our sound bank(s). I named my event *Impact_Cube*, dragging and dropping the *Cube_Collision* SFX object into the empty space near the top.

c.PNG

Next, we'll create a blueprint for posting the sound upon collision with our desired object. In the scenario above, it will be one of these blue cubes. If you want to quickly test out this Wwise configuration, I am using the 3rd Person Demo scene that Unreal provides its users for free as a starting template for project creation.

d.PNG

In the right view, I have the details panel open. By clicking the blueprint icon in the top right, the window in the middle opens up with options for creating a blueprint. We want to create a new subclass and for this blueprint to be tied to the StaticMeshActor, since the mesh is what the player can collide with.

For clarity's sake in a developing project, I prefer to start the names of blueprints or scripts that are related to Wwise with "Wwise", alowing me to easily distinguish which components are for audio purposes. I named my Blueprint *Wwise_CubeCollision.* The next thing to do is open the details panel and check for the following options to be selected:

- Physics > **Simulate Physics**

- Collision > **Simulation Generates Hit Events**

e.PNG

If you're in the view above and do not see the components tab, go to the top toolbar and Click **Window --> Components** to open the Components tab.

f.PNG

Right click the Static Mesh Component in the Components View and click **Add Event --> On Component Hit**. This will open up the Blueprint Editor, showing you the added component.

image.png

By right-clicking empty spaces in this view, adding and connecting components to one another, you need to configure the setup above. If you're having trouble finding certain components, here are some tips:

- The grey box, a greater than or equal to if statement, is found by typing ">=" in your search for a new component to add.

- If you cannot find your RTPC in the Wwise drop down, try opening the Picker Tab window, then dragging and dropping your RTPC from the Game Parameters Folder. If you do not see the RTPC in your Game Parameters folder in Unity, save your project in Wwise, make sure the event was dragged into the sound bank upon generating it in the sound banks view, then save your project again and allow Unreal to reparse before checking to see if the RTPC is now showing up.

- You can use the drag and drop method from the Wwise picker tab into any component asking for a reference to an object in Wwise that is needed. This goes for the Post Event Component!

To Summarize what this Blueprint is doing, I will be echoing what the linked website at the beginning tells us:

1.) It detects when the object collides with another.

2.) It sets the value of the RTPC based on the Vector Length of the Component Velocity, and ensures that the value exceeds a minimum threshold of 40, which ensures that minor contacts do not cause impact sounds.

3.) It posts the Event to Wwise with the RTPC value, so that Wwise plays the impact sound at the appropriate volume.

g.PNG

The blueprint is complete and you can now click Compile in the top left.

image.png

A more fun way to test the collision effect is by giving you the ability to spawn cubes during playmode! Here are some tips for finding these components during your search.

You can reference a quick 1 minute video tutorial, linked here.

- The grey x, our multiplication component, can be found by typing Multiply and selecting *Multiply* under the Operators section. You can find the addition component by typing '+'.

- The dot in the grey box is a converter. It will convert a value from 'this' to 'that' and there are many types of conversions that can be made. To know what type of conversion you're doing, the pins are colored and represent a type of value. In this case, the conversion is from a Vector to a Transform value. You can make this box appear by dragging the orange pin next to 'Spawn Transform' to the upper right yellow pin in the + node, or you can look up 'Vector to Transform' and should find the component.

- F is a *keyboard event*, so when you type F in your search, look for the keyboard event section and select the one that says 'F'.

- You may have noticed the pin in the bottom left of the x component is green, whereas yours starts off yellow. You can convert this pin by right clicking it and selecting **Convert --> Float (Double-Precision)**.

- "SpawnActor Wwise Object" is not what you should type, when searching for this component. You only need SpawnActor because what comes next is the name of the gameObject that will be spawned.

Revision #2
Created 2023-11-13 14:28:28 UTC by BaggoNotes
Updated 2023-11-13 15:32:56 UTC by BaggoNotes