DID IT HIT PLUGIN

Project Zero, 2021

Contents

Scope and purpose

The plugin was designed to work with fast or low framed movements where the detection based on overlap may sometimes falter. For instance, if you have an RPG and you swing your sword, the frames at

which the swing happens may skip a crucial position and render that swing useless. Another scenario are small projectiles that move too fast against thin targets.

All of this can be resolved through trace methods that take in the initial and final position each frame.

The main scope of this plugin is to cover these gaps in collisions. Currently it is using the kismet methods however another method will be implemented in the future as well and given the flexibility to the user.

As such, the plugin will do 3 different traces:

1. A trace between the same point but at different times.
2. A trace between all different points at the same time.
3. A trace between all different points at different times.

Based on testing, this is overkill and you as a user may choose which trace is most suitable. For instance, the first method is perfect for bullet; whereas the other 2 methods are inadequate.

Setting Up

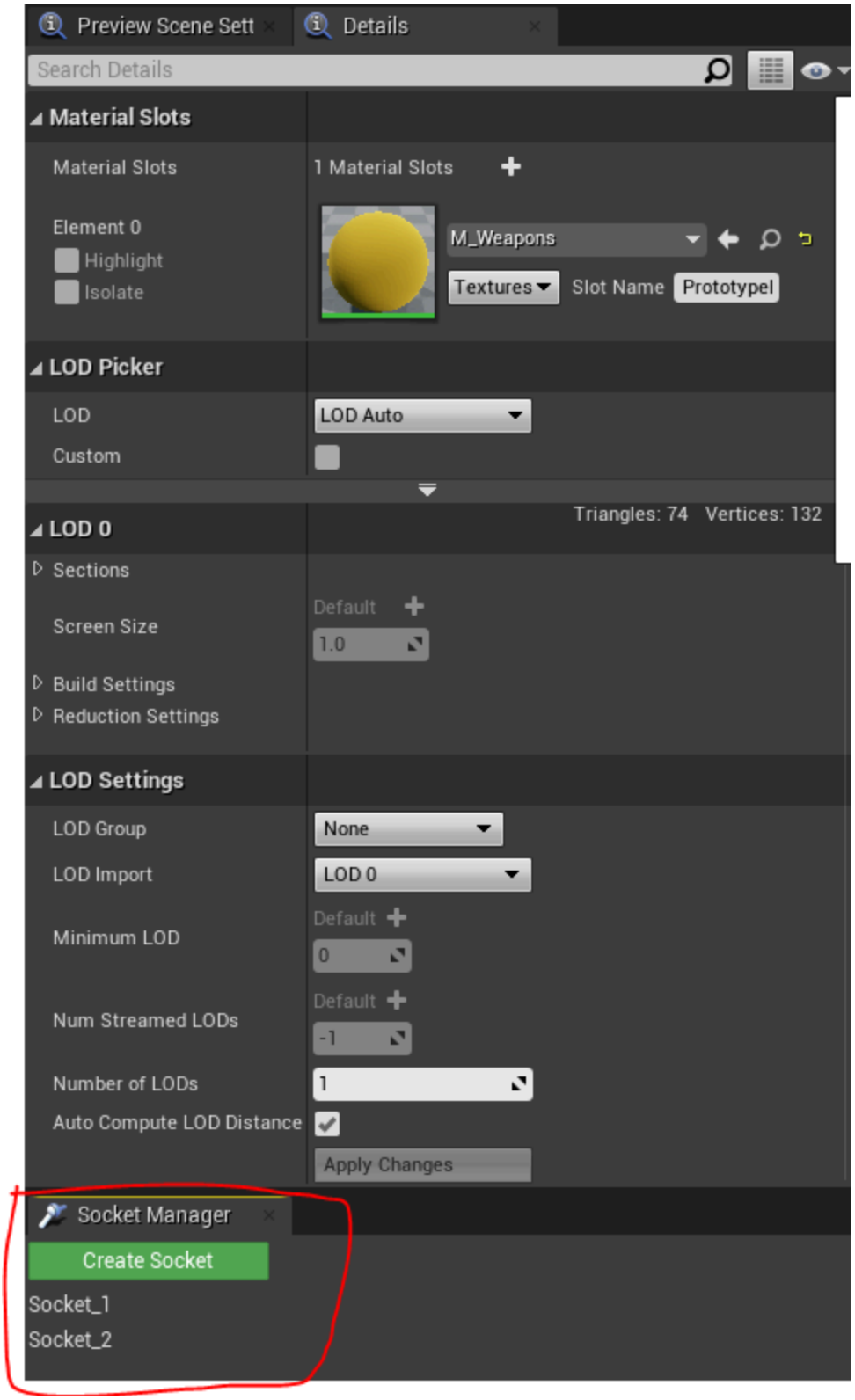
The plugin requires 3 crucial setup steps:

1. Adding sockets to the static mesh.
2. Adding the actor component either on the projectile or the character.
3. Defining the static mesh component to the actor component.

Adding sockets to the static mesh

The sockets will be the points of reference that the plugin will use to determine the extremities of the tracing object. You can add any amount between 1 and infinity although, more than 20 may result in performance issues. **You can name your sockets anything as long as all socket names are different.** To do so:

1. Open your static mesh
2. Click on “Add socket”



3. Move your socket on the model as needed.
4. Repeat steps 2 and 3 as needed. Some weapons may require more than 2 sockets to improve precision.

5. Save your static mesh.

Adding the actor component to either the projectile or the character

As of 5.1, the actor component is called "AC_DidItHit"; all versions prior it is called "AC_DitItHit". This typo was discovered late into development and after testing, changing this in previous versions will force users to rewrite all their code with this plugin. Moving forward starting in 5.1, this will be correc.

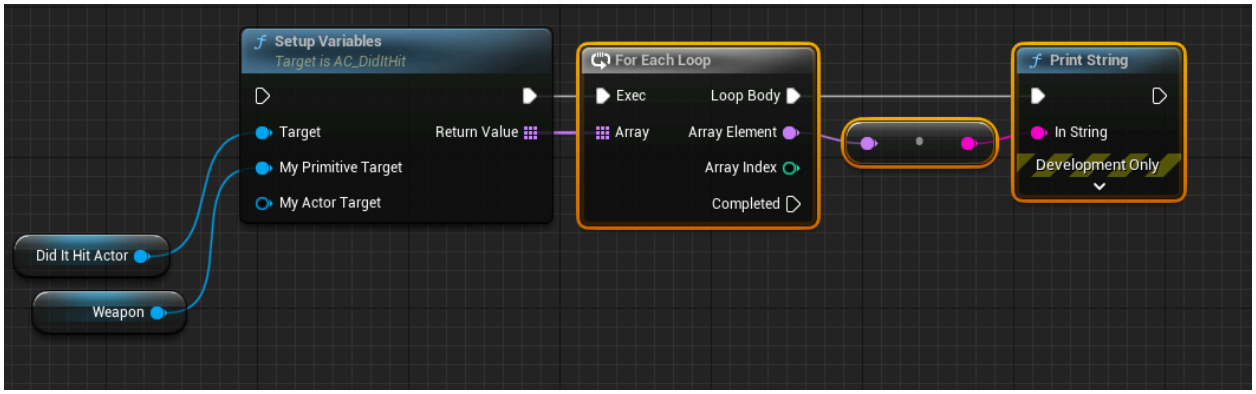
If you have a projectile, the assumption is that your static mesh will have 1 socket at least. Simply add the actor component in the viewport. It will appear as AC_DidItHit but the name will change once it is loaded in.

If you have a character with a weapon child or static mesh, add the component to the character.

Defining the static mesh component to the actor component

In the construction script for the actor component, drag and drop the "Did it hit actor" component and drag off of it and look for Setup Variables. This function will need either a primitive target (the static mesh component) or an actor component. If none are provided, it will default to the owner of the actor component.

This helps the actor component in searching for all available sockets in the static mesh.



If you intend to change weapons midgame, rerun the setup variables again for the new static mesh. In the example project, 3 weapons are provided and swapping them calls this function each time.

UPDATE: A return name array was added to debug which sockets were setup. If this returns nothing, it means your sockets were not found.

Older example projects:

LINK (4.26) use key: uyTtd1BekhBSO1EcjHZ5XY3n1lpCVbFvl_o3IQAWgxE

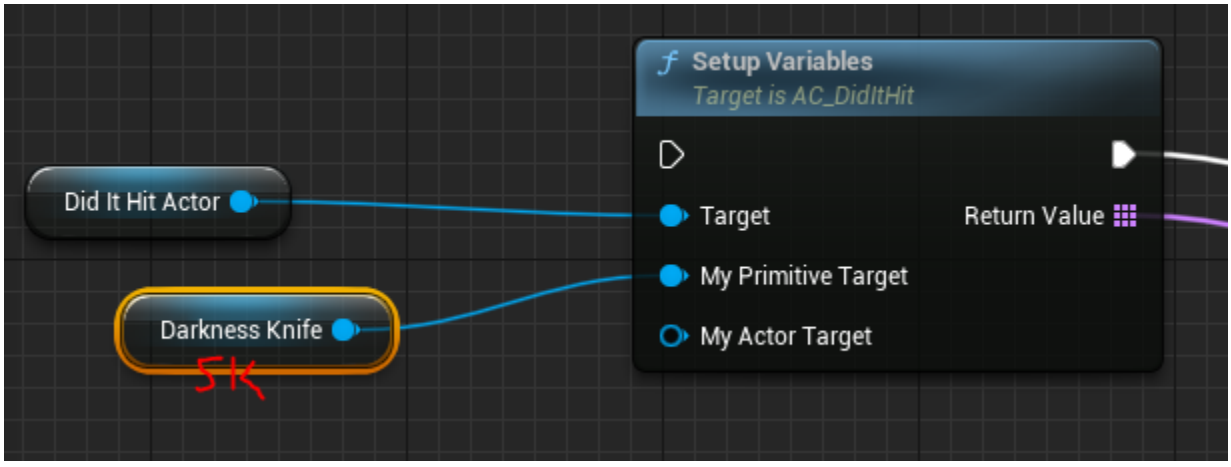
5.0 Link: use key: nv2qtLx9qU48JFXCr5IW5UOwxqBcYnW4xybySuBJQos

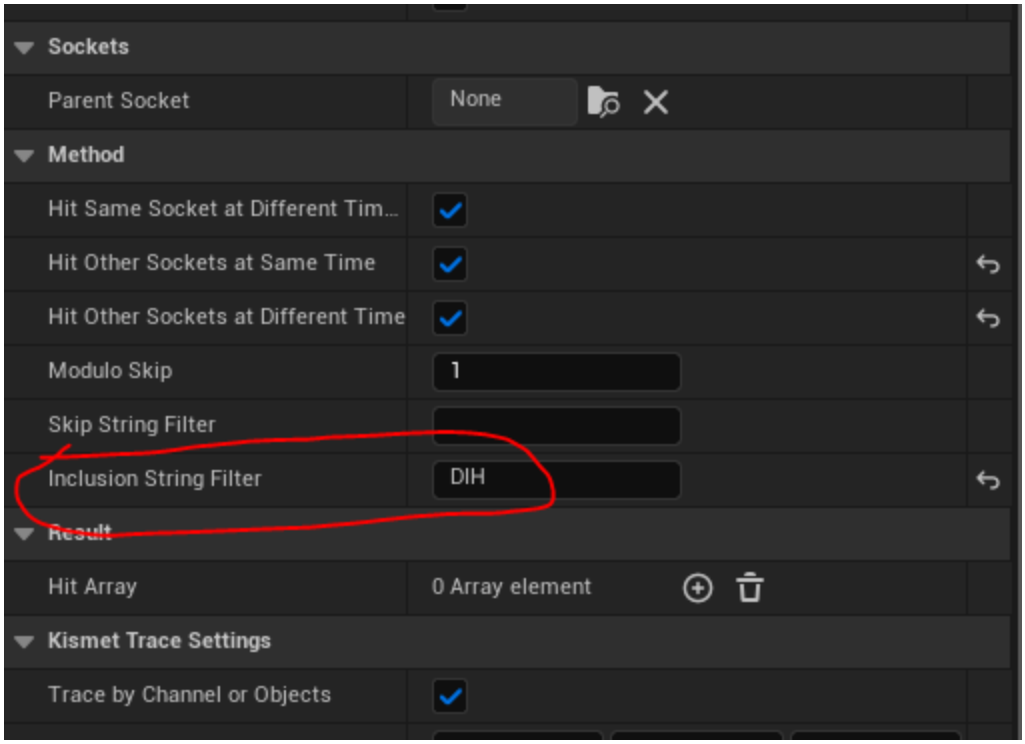
Using Skeletal Mesh Sockets

In 5.1+, skeletal meshes bones and sockets are all considered under the same umbrella as sockets. As such, you can technically use skeletal meshes. The problem is that if you set it up as a variable, all bones will be used.

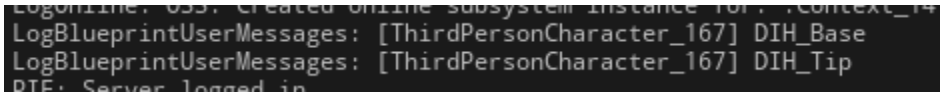
To remedy that, use the String Inclusion filter to include the bones/sockets you want to use:

The skeletal mesh must be used a primitive explicitly. You cannot use the Actor target for this.




Sockets
Parent Socket
None
Method
Hit Same Socket at Different Tim...
Hit Other Sockets at Same Time
Hit Other Sockets at Different Time
Modulo Skip
1
Skip String Filter
Inclusion String Filter
DIH
Result
Hit Array
0 Array element
Kismet Trace Settings
Trace by Channel or Objects


Search Skeleton Tree...
Name
Root_Bone1
Grip_Bone
DIH_Tip
DIH_Base
Perspective
Lit
Show
Character
LOD Auto
x1.0
Previewing Reference Pose
The following materials need morph support ('Used with Morph Targets' in material editor):
/Game/SciFiWeapDark/Weapons/Materials/M_WeaponMaster_Em.M_WeaponMaster_Em
LOD: 0
Current Screen Size: 1.683
Triangles: 7,056
Vertices: 6,247
UV Channels: 1
Approx Size: 8x46x55


LogBlueprintUserMessages: [ThirdPersonCharacter_167] DIH_Base
LogBlueprintUserMessages: [ThirdPersonCharacter_167] DIH_Tip

Did It Hit Actor Component Parameters

The actor component contains several parameters to play with. The first is which of the 3 methods you wish to use. By default the line trace for start and end of the same socket is enabled however, you can enable all 3 at once if you'd like!

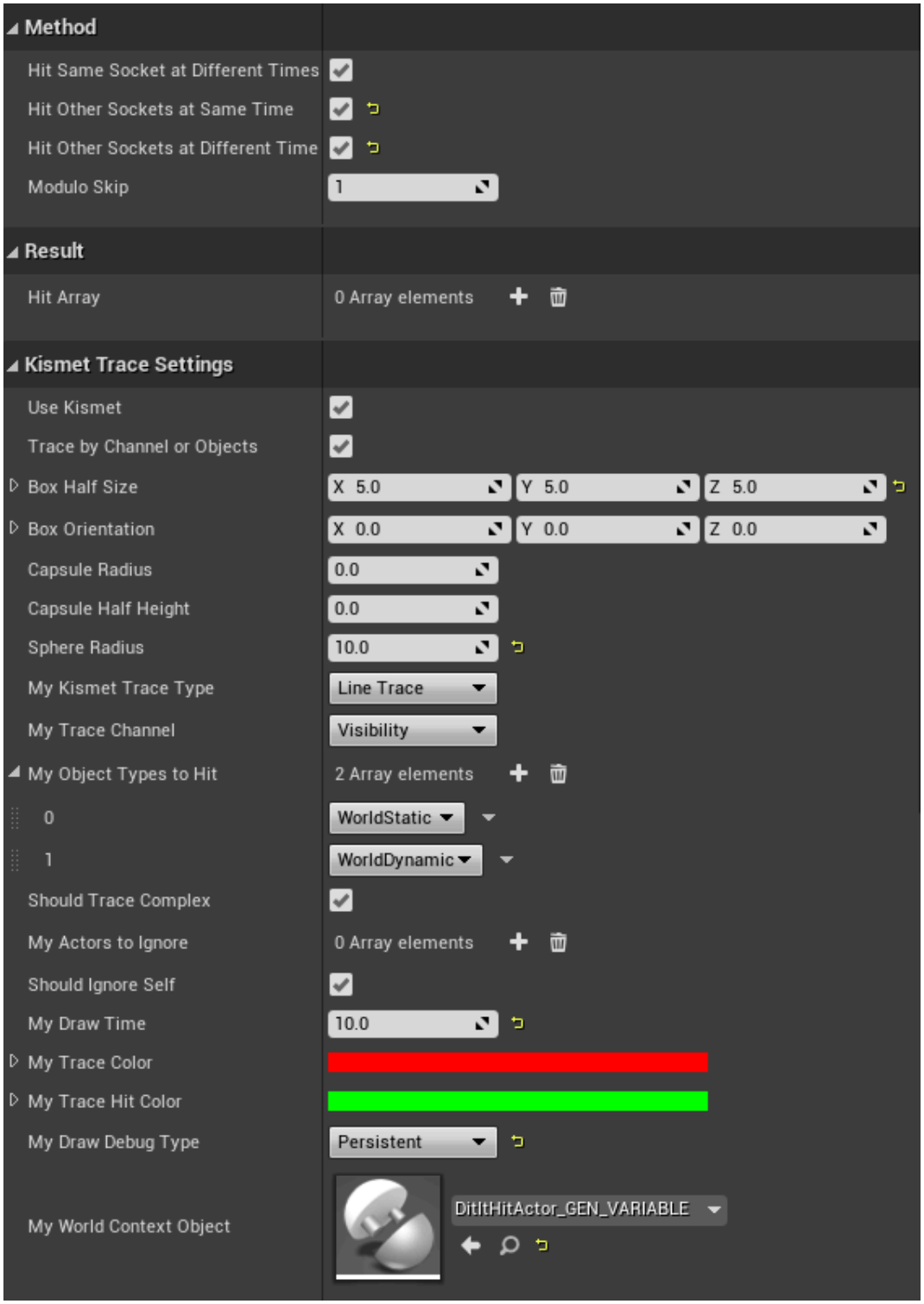
A modulo method was also created if you do not want to run this on every single tick but every other tick. The integer will reset on toggling the trace on and it will only tick every n^{th} time where n is the modulo skip integer.

The hit array is also visible though other than getting the results, you won't be needing to meddle with it too much.

The main methods are all derivatives of the Kismet methods that you find in the blueprint (think linetrace for instance) and all parameters for that method have been made public. Currently, only the Kismet method is available and as such you MUST set it to True. Otherwise nothing happens.

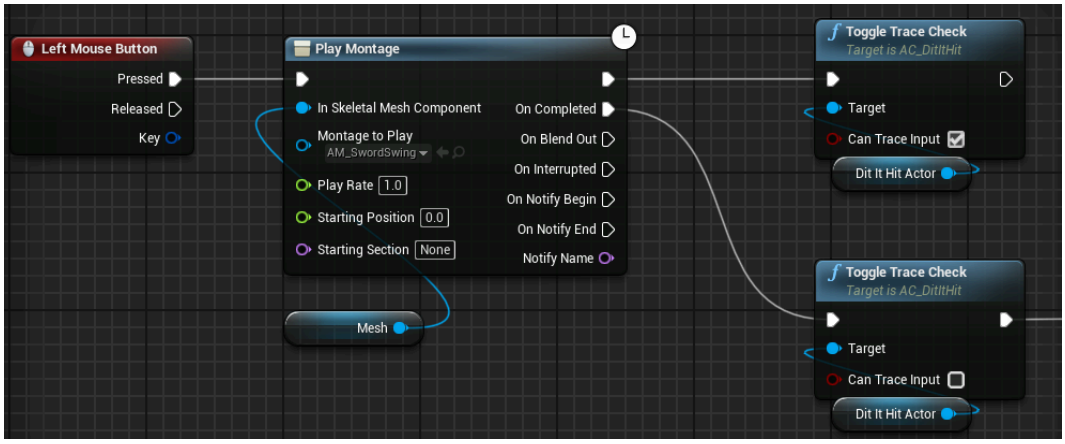
Note for the TraceByChannelOrObjects boolean (2nd Kismet trace setting):

True means you will trace by channel and false means you will trace for objects. If you set it to false, you need to have object types for the tracing to work.


Method
Hit Same Socket at Different Times
Hit Other Sockets at Same Time
Hit Other Sockets at Different Time
Modulo Skip
1
Result
Hit Array
0 Array elements
Kismet Trace Settings
Use Kismet
Trace by Channel or Objects
Box Half Size
X 5.0
Y 5.0
Z 5.0
Box Orientation
X 0.0
Y 0.0
Z 0.0
Capsule Radius
0.0
Capsule Half Height
0.0
Sphere Radius
10.0
My Kismet Trace Type
Line Trace
My Trace Channel
Visibility
My Object Types to Hit
2 Array elements
0
WorldStatic
1
WorldDynamic
Should Trace Complex
My Actors to Ignore
0 Array elements
Should Ignore Self
My Draw Time
10.0
My Trace Color
My Trace Hit Color
My Draw Debug Type
Persistent
My World Context Object
DitItHitActor_GEN_VARIABLE

Working example

In the example project,a sword and a swing animation are provided for testing purposes. When using the left mouse button, you will trigger a swing which will toggle a boolean to true. The actor component will then start tracing on tick. Once the animation ends, it will toggle it back to false thus stopping the trace.



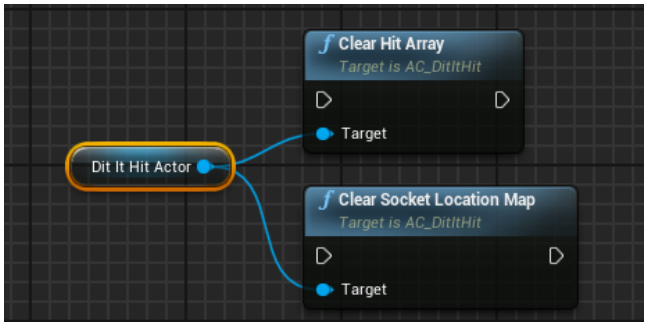
Further ahead, after the swing is done, the HitArray is then called directly from the actor component. Admittedly this would work much better with anim notifies however it is out of scope.

In the example project, all this does is print string the name of whatever was hit. You can push it further and apply damage or effects or destroy them.

Additionally a projectile on the right mouse button was added to aim and hit wherever the camera is pointing. You can change its velocity in the ThirdPersonCharacter blueprint on spawn.

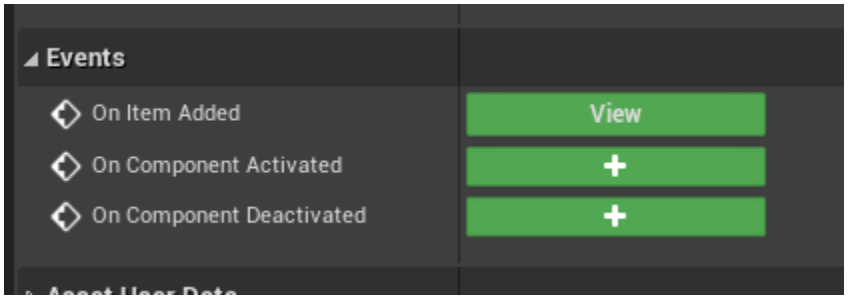
As the speed of the projectile increases, the higher the chances of it skipping collisions. However, if the actor component is active, it will calculate between positions if something was hit.

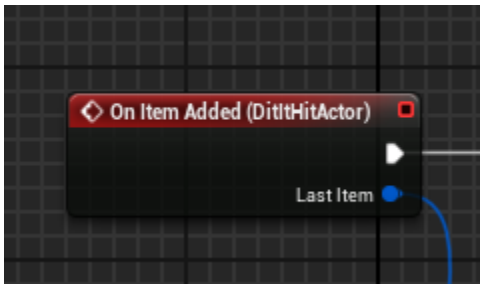
To allow the user more control, these two functions were made available. They do exactly what they say…



NEW FUNCTION ADDED!

A new event called “OnItemAdded” was added to the actor component that can be called on the character. You can simply **select the actor component**, and there will be a new event available on the description at the bottom of the right hand panel:





What this event does is that when that DiH actor component adds any item/hit to its array, it will trigger and you can then get that last item added.

Example: you swing a sword, the traces are enabled, during a trace you hit an object mid swing: immediately when that happens, the OnItemAdded event will trigger and you can use it to inflict damage, play a VFX/SFX with that last item in focus. The item is a hit struct so you can break it into components to either use the actor, the location, etc.

Future Plans

Currently only the Kismet methods are used. In the future, a WorldTrace function will be implemented that is technically more performant. By how much is another question. Preliminary tests have shown that the increase in performance is minimal.