Trailmarker Manual

The 'Trailmarker' is a tool for leaving marks of objects on the ground.
Developed to be versatile, fully dynamic, optimized and easy to use.

Table of Contents

Features

Project markings on landscapes, procedural, static or skeletal meshes.
Add markings to a decal or your material through a material function.
Made with optimization in mind:
Only capture when needed or told to.
Automatically release render targets from memory.
Features not used can be turned off to save performance.

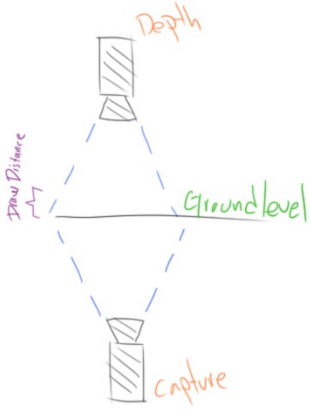
Activated by player pawn, chosen classes or actors that are inside customizable bounds.
Support motion and rotation.
Save and Load to disk.

Fundamentals of leaving a mark

...with this tool, on the ground in your scene.

Concept

Capture the silhouette of an object from below and using the depth information of the surface to mask it within the draw distance. Adding the silhouette from the capture render target like it would be a brush to another render target that would be the canvas, saving all markings on it until it's cleared.



Any object class that can generate overlap events may activate the scene capture components when entering the inner bounds, an area of which the markings can be captured and displayed in. The outer bounds may clear the canvas, save and load.

There are two ways markings can be displayed on a surface, from a decal or inside the material with the help of a material function. The world-aligned method of the material function allow each material to handle the markings in their own way as a mask inside it.

Introduction

Basic Setup with a Decal material

1. Place Trailmarker actor in the scene.

2. Scene capture and bounds will automatically change when the 3d widget 'Actor Search Location' is inside an actor's bound, and a valid 'main actor' is found.

   Main actor can also be set in Settings/Capture, 'Override Main Actor'.

3. If other actors should receive markings, add them to 'Additional Actors' in Settings. Actors that shouldn't receive markings, add them to 'Actors Hidden from Capture,' to avoid that they're leaving markings on themselves.

4. Disable 'Receives Decals' (Details / Rendering) on actors that shouldn't receive this decal but will be inside its volume.

5. The Decal volume (green box) will inherit the size from the orthographic width. It can be edited in height from the Decal Settings category in the Details panel.

   The surface receiving the decal projection should be at the center of the decal's volume.

6. 'Outer Bounds' will decide when the render target displaying the markings on the ground should be cleared or saved / loaded. Scale it to fit your scene.

   Check 'Overlap from Outer Bounds' to preview outer bounds changes while editing the 'Scale Outer Bounds' Don't forget to uncheck the box when done, (if it's not the desired function).

7. In Activation Settings, change Activation Method to Specified Classes. Add actor class and component class to the 'Filter from Classes' map structure. Now more types of objects can activate this marker. Not only the player pawn.

   The component must have a collision and generate an overlap event to work as a trigger.
   Filter from classes Example: Actor: ThirdPersonCharacter , Component: CapsuleComponent

8. Edit or make a new Decal material with the necessary functionality to be compatible with the Trailmarker and your scene.

   Basic Setup Complete!

   Tip: Check the 'Hide Components' default variable to hide visual aid when the setup is done.

About Projection: Decal

Decal projection is probably less demanding from a performance perspective, and it uses only one material. It can move and rotate at runtime without updating any parameters. Decal projection is very dependent on its volume and where the receiving surface is in comparison to it.

Introduction

Basic Setup with Material Function

1. Add a Trailmarker material function in all of the materials that should receive markings. It will output a mask and UV coordinates to use when adding markings to the material.

   (See: 'Material Function's and how to use them')

   The component receiving the projection cannot share the same Material Function in one material with another Trailmarker. It's possible to add more than one material function to the material if custom parameters are assigned.

2. Place the Trailmarker actor in the scene.

3. Scene capture and bounds will automatically change when the 3d widget 'Actor Search Location' is inside an actor's bound, and a valid 'Main Actor' is found.

   The 'main actor' can also be set in Settings/Capture, Override 'Main Actor.'

4. Change Projection in Settings to 'Material Function.'

5. If other actors that should receive markings, add them to 'Additional Actors.'

6. 'Outer Bounds' will decide when the render target displaying the markings on the ground should be cleared or saved / loaded. Scale it to fit your scene.

   Check 'Overlap from Outer Bounds' to preview outer bounds changes while editing the 'Scale Outer Bounds' Don't forget to uncheck the box when done, (if it's not the desired function, recommend not using this option in game).

7. In Activation Settings, change Activation Method to Specified Classes. Add actor class and component class to the 'Filter from Classes' map structure. Now more types of objects can activate this marker. Not only the player pawn.

   The component must have a collision and generate an overlap event to work as a trigger.
   Filter from classes Example: Actor: ThirdPersonCharacter , Component: CapsuleComponent

   Basic Setup Complete!

   Tip: Check the 'Hide Components' default variable to hide visual aid when the setup is done.

About Projection: Material Function

Material Function is probably a bit more demanding on performance than using a decal but offers more functionality. The material function is 'aligned to the world position,' and outputs a mask and UV coordinates to the material. Multiple 'Trailmarkers' cannot share the same material function, but they can share one material on the same component if it has another material function with custom parameters assigned to it.

Material Function's

TrailmarkerDecal

Material function for the decal in the 'Trailmarker'.
Output a mask ('Trailmarkings') and UV coordinates.

Mask based on the render target from the 'Trailmarker' actor, use it in the decal components material.

Input

ProjectionMaskZ (B).
Show decal projection only on surfaces facing up in the world direction.
Connect a constant with the value of zero to disable this function.

SlopeMask: **SlopeAngle (V3)** *"World direction for slop, typically points down"*
Slope Falloff (S) *Falloff Power*
Cheap Contrast (S)

Note: "Decal projection in the 'Trailmarker' actor, using the default parameter names, because the decal has its material, there is no need to change any."

Decal Projection lets the 'Trailmarker' actor move without having to update any parameters.

TrailmarkerWorldAligned

Material function for surfaces that should be able to receive markings. The material function doesn't need any custom input to work with the default parameter names, and it will Output a mask ('Trailmarkings') and UV coordinates.

TrailmarkerWorldAligned can only work with one 'Trailmarker' at a time. When more than one 'Trailmarker' shares the same surface and uses the same material instance. Each 'Trailmarker' must have its material function.
(More information on Page 6)

Input

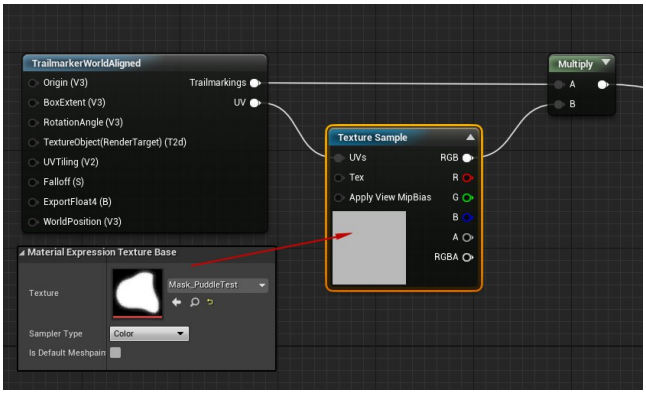
Parameters sent from the 'Trailmarker's blueprint.

Origin(V3) *'Trailmarker' decal component world origin.*
BoxExtent(V3) *Box Extent based on orthographic width.*
RotationAngle(V3) *Rotation Angle 0-1.*
TextureObject(RenderTarget)(T2d) *The Render Target that is going to be displayed on the surface.*

UVTiling(V2) *It's possible to edit the UV Tiling, but it shouldn't be necessary.*
Falloff(S) *Falloff based on the world-aligned parameters, if not masking with a texture.*
ExportFloat4(B) *Change output to a float 4, not tested and shouldn't be necessary.*
WorldPosition(V3) *World position with no material shader offsets applied.*

To move the Trailmarker around at runtime, it must have 'Enable Motion Support' checked in Settings.
Updating the origin and rotation angle on Tick event.

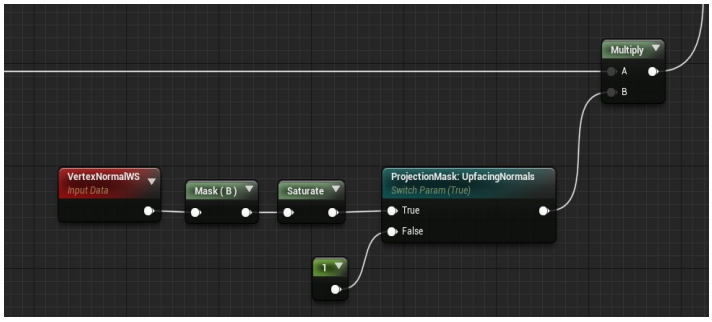
About the UV coordinates output



UV coordinates output in these functions can make additional masking on top of the render target. It can be for various purposes, like shaping the markings canvas, add a pattern, or applying a falloff.

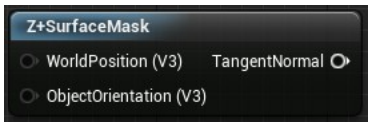
(See Image →)

Up-facing Normal mask



Material using the Trailmarker Material function would have to display its result only on surfaces facing upwards (Z+).

(See Image →)



Z+SurfaceMask

It will mask any rendered surface normals in the Z-axis. Useful when the masking has to occur outside of the material influence. Displaying the decal on top, but not in any other axis.

Moving platform

Apply motion to the 'Trailmarker'.

To get markings on a moving platform or surface.
The 'Trailmarker' needs to follow the platform at its angular and linear speed.
Decal projection will follow along nicely, but any material that is world-aligned have to get a new origin and rotation value for each frame. This function isn't activated on default for optimization reasons.

Enable Motion Support

Enable Motion Support in 'Details / Settings / Capture.'
Allows the 'main' actor's components material that is going to receive markings. (/projection) to update its origin and rotation angle for each frame (at tick event).

Enable Motion Support on Additional Actors

Enable Motion Support is only supporting the main actors components.
To use it on Additional Actors as well: 'Motion Support On All Actors' * must be checked.
* *Hidden variable*

Attach 'Trailmarker' to another actor.

It can be attached to another actor, be careful not change its world rotation and scale.

'Trailmarker' can be added as a child actor component inside another actor.

When the 'Trailmarker' is a child actor, add the 'TrailmarkerSettings_BPC' actor component to the parent actor, for easy access to the 'Trailmarkers' settings from the parent actor details panel.

Tip: *When using the 'Trailmarker' as a child, some visual aid won't show any change in the editor. Recommend using another 'Trailmarker' actor directly in the scene at the same place, to see the scene capture cameras location and volumes such as decal volume or the inner / outer bounds displayed by the overlap collision box.*

Restrictions and working around them

Layers

The 'Trailmarker' can't capture in layers. It can only capture one surface at a time, between the scene capture 2d cameras. If two objects are both added to receive markings on top of each other from one 'Trailmarker,' the surface on top will decide where the ground level is, and they will both share and display the same markings.

A Solution to this problem is to place two different 'Trailmarkers' in the scene and adjust the one on top so it can't capture anything that happens on the bottom.

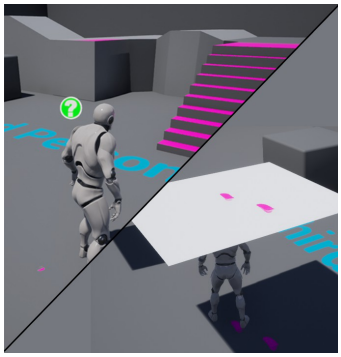
Object below ground level are leaving marks on the surface

Visible objects below the ground level but in front of the 'Trail marker Scene Capture 2d' may get captured, and if it does, the entire actor will add markings on the surface.

There are a couple of actions that can prevent this.

1. Place the scene capture in front of the object that is obstructing the view if possible.
2. Add the obstructing actor to the 'Actors Hidden from Capture' array in Settings.
3. For Decal Projection: Go into the actor's details panel and uncheck 'Receives Decals.'
4. Go inside the 'scene capture 2d,' and uncheck the obstructing object class from being rendered.
5. Set 'Below Surface Level Culling' to true in Settings.*

**Overlapping actors are hidden from the scene capture if they are below the surface level. This feature depends on the inner bounds to set up correctly and the actors to have generate overlap events enabled.* *(More information on page 7)*



The bottom faces are being rendered by the scene capture on the image above, left side.
It can be added to Actor Hidden from Capture or Additional Actors in Settings.

Decal

Actor Hidden from Capture still receive decal projection

The decal is projecting on surfaces that shouldn't receive any. It's a problem because it's not about hiding them from the capture, rather than them displaying the decal projection that has already been from another surface below them. *(← image, right side)*

A solution to this would be not to have the decal volume overlapping any other object than the surface that should receive them. Another way to solve this problem would be to uncheck 'Receives Decals' in the details tab of the actors, or inside the material, go to the Decal Response (D-buffer) and set it to None in the drop-down menu.

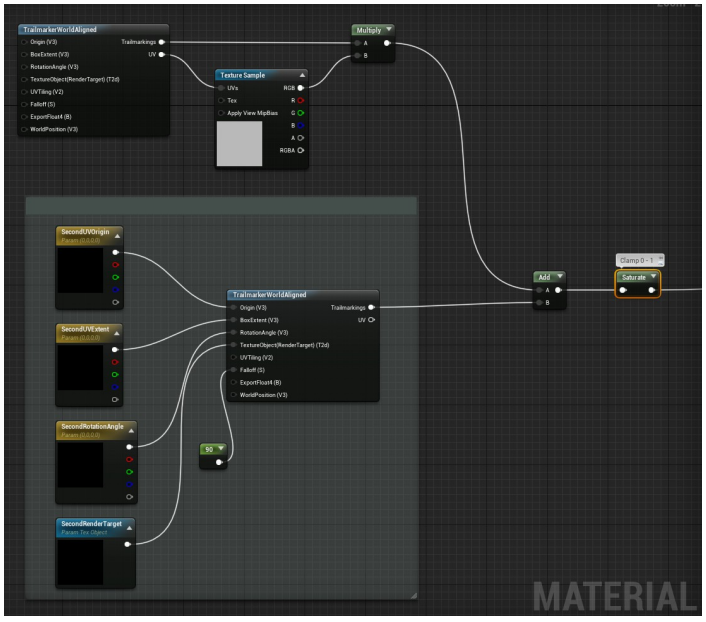
Material Function

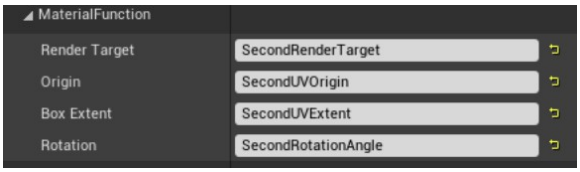
'Trailmarkers' cannot share the same material function.

If two or more 'Trailmarkers' are placed on one surface and using the same material instance. They would send different parameter values to the same material function. It results in flip-flopping between them.

The solution is to give each 'Trailmarker' its own material function and unique parameters.

(← See Image)





Assign new parameters by name in the 'Trailmarker'
(Details panel: Settings / Material / Material Function).

Below Surface Level Culling

Below Surface Level Culling is automatically hiding actors* below the surface level from the scene capture 2D. There are two variants of Below Surface Level Culling.

** An actor and/or its components must be able to generate overlap events.*

Below Surface Level Culling

This is a simple and “cheap” version.
It consider the surface level to be in the middle of the two scene capture components.

Below surface level are only checking if any overlapping actors are below or above the surface when they begin to overlap.* It's important to set up the inner bounds *(if not updating on tick)*, to be sure that all overlapping objects only can transition between the surface levels when they are outside the overlap area/ inner bounds.

This is because it's only updating an actors visibility to the scene capture component when the actor begin overlap.

***There's a hidden continuous version of this feature that check all overlapping actors on tick. It's a hidden variable that can be used instead.*

Below Depth Level Culling

Advanced version of the Below Surface Level Culling, that's reading pixels from the scene depth taken by a scene capture 2d component, and compare them with the overlapping component.

Below Depth Level Culling have two hidden tolerance variables for enter and leaving an area from under a surface that can receive markings. If your actor is bleeding over to the surface markings when going in under or leaving, these two can be tweaked.

Below depth level culling can be heavy on performance, please take that in consideration when implementing this feature.

Multiple Materials

The 'Trailmarker' can display markings on multiple materials on the same mesh. 'Material Slots' variable have an array to add material slots to. It will only assign new dynamic material instances to available slots, and treat any scene component the same, except if that scene component is ignored by the tag included in the material slots structure variable.

Multiple materials doesn't work on landscapes, use Landscape Blend Layers instead.

Landscape

'Trailmarkers' can be used on landscapes as well,
but the landscapes must have 'Use Dynamic Material Instance' set to True.

Recommend using Tags on Landscape components.

When a landscape is assigned as the main or additional actor it will create new dynamic material instances for all components on that landscape. If not a tag is assigned on the components and added to the Trailmarker settings attribute 'Components with Tag' (Details/Settings/Capture/Landscape). If using multiple 'Trailmarkers' on one landscape, it's recommended to use tags to separate them, only the 'Trailmarkers' sharing the same component have to share the same materials.

Landscape Blend Layers

When blending materials on a landscape, it will be a falloff on the material, any falloff settings to the 'Trailmarker Material Function' won't be needed.

Save Load and Clear

'Trailmarkers' can save, load, and clear the render target from a file.
In Settings, the Render Target Manager (Settings/Capture/Save and Load) can activate the save and load features when the first or last triggering actor enters or exit the outer bounds. The save file for each 'Trailmarker' will be unique for the scene they are in and, the directory can change in settings, with the base directory selection and directory path.

Save, Load and Clear functions can be called directly with a cast or blueprint interface, (not included).

Save

Save Render Target to a texture .png file on the disk.
Target directory path can be chosen by two variables in Settings.
File name cannot be chosen, it's based on the level and 'Trailmarker' name and id.

Load

Load from an existing file in chosen directory.
Loading a file with no triggering actor inside its overlapping bounds will still load and projecting markings on the surface. Not to be unloaded from the memory until an actor that is specified as a trigger has entered the overlapped section, and then exit the outer bounds.

Clear

Clearing the content of a saved file on the disk can only be done through a function.

Note: *Trailmarker doesn't include a delete file function.*
Exported render target won't be deleted and left on disk.
See page 17 – for a short guide on how to solve this with a 3rd party code plugin.

Settings Encyclopedia [Part I]

Explanation of all variables that can be set in the Settings details category.
In the editor each variable has its own explanation to it, if not obvious what it does from the title.

Settings

Override Main Actor	Set your own main actor and disable 'Find Nearest Actor' function.
Main Actor	Only for preview currently selected main actor.
Material Slots	The receiving main actor material slots. Add material slots that should receive parameter values from the Trailmarker to its material function.
Additional Actors	Additional actors other then the main actor to receive markings on their surfaces. Including material slot settings for each additional actor.
Actors Hidden From Capture	Hide actors that shouldn't get markings on their surfaces.
Projection	It's the core method of how the markings will be displayed on a surface.
OrthographicWidth	The size of the area in the scene which the markings will be displayed.
Hide Components	Hide components that's only visible for visual aid on setup.

Settings|Capture

Capture on Movement	Only run Scene Capture when specified actors are moving inside the inner bounds.
Movement Tolerance	Set tolerance of how fast a specified actor must travel to start the capture.
Bounds	Edit decal volume, inner and outer bounds by using different methods, Box, Plane and Manual. For the scene capture, Box and Plane are using the main actors bounds to set the height of the scene capture components. Manual is using the 'Trailmarkers' pivot location instead.
Capture Height Offset	Offset scene capture components location in Z.
Scale Inner Bounds	If not overridden the inner bounds will handle on and off features of the scene capture components and specified actors that will activate the 'Trailmarker' when they overlap.
Scale Outer Bounds	Outer bounds can save, load or clear the render target that's projected on the surface. Scale it to be far from the inner bounds as it will show if the Manage Render Target is set to 'Clear' or 'Save and Load', and the last actor that is specified to work as a trigger would enter or exit.
Overlap From Outer Bounds	Override the inner bounds by setting the 'Overlap Detection Box' to the outer bounds. Removing the inner bounds functionality. This can be good when scaling the outer bounds as it will show where it's in the editor, but don't forget to uncheck this box after its done. If it isn't desired to run both scene capture and their render targets from the outer bounds.
Enable Motion Support	Let the 'Trailmarker' move and rotate at runtime when projecting to dynamic material instances that are using the 'Trailmarker' material function.
Motion Support On All Actors*	Let the 'Trailmarker' move and rotate at runtime and projecting on additional actors too.
Enable Rotation*	This is enabled at default as it's a part of the 'Enable Motion Support' functionality. If a platform should be moving but not rotating, this can be turned off.

** Hidden variables*

Settings Encyclopedia [Part II]

Settings|Capture|Below Surface Level Culling

Below Depth Level Culling	If any actor when begin overlap is below surface level, hide it from the scene capture. It's reading the Scene depth. Important: This feature may be slow/ heavy on performance.
Above Depth Level Tolerance*	Multiply search radius when going from above a surface to below, reduce bleeding.
Below Depth Level Tolerance*	Multiply search radius when going from below a surface to above, reduce bleeding.
Below Surface Level Culling	If any actor when begin overlap is below surface level, hide it from the scene capture.
Below Surface Level Culling (Tick)	If any overlapping actor is below surface level, hide it from the scene capture.

Settings|Capture|Save And Load

Manage Render Target	Chose how the 'Trailmarker' will handle the render target when the first or last specified actor enter or leave the outer bounds. Clear, Keep in memory or Save and Load from file.
Load On Begin Play	Load from file on begin play.
Clear On Begin Play*	This will clear the save file from data on begin play.
Override Loading Manager*	Load on begin play even if the Manager Render Target is not 'Save and Load.'
Base Directory*	Chose between project content folder, project save folder or no base directory path.
Save Directory*	Chose a directory relative to the base directory, to save, load and clear files from.

Settings|Capture|Landscape

Components With Tag	If a tag is specified, 'Trailmarker' can only use landscape components with that tag.

Settings|Capture|Decal

On Begin Overlap Disable Receives Decals	Automatically disable receives decals on overlapping actors.
On End Overlap Enable Receives Decals	Automatically enable receives decals on overlapping actors.
bIncludeChildrenOnReceivesDecals*	If receives decals is being changed on overlapping actors, include their children as well. (version no. 1.00.4)

** Hidden variables*

Settings Encyclopedia [Part III]

Settings|Material

Image Resolution	Image resolution of the actors render targets. (Don't go crazy with this one)
Draw Distance	Draw Distance calculated from the surface that is receiving the projection and up to set height. Between the surface and the draw distance height, components will get captured, anything above the draw distance height won't be.
Contrast	Contrast(Opacity) of the markings projecting on the surface.
Auto Generate Mip Maps	Auto Generate Mip Maps for all Render Targets

Settings|Material|Decal

Volume Height Offset	Scale capture area in Z axis (for Plane and Manual bounds)
Volume Height Position Offset	Offset location for capture area in Z axis

Settings|Material|Material Function

Render Target	Parameter name for the material function.
Origin	Parameter name for the material function.
Box Extent	Parameter name for the material function.
Rotation	Parameter name for the material function.

(More information on page 6 - Material Function)

Settings Encyclopedia [Part IV]

Settings|Activation

Activation Method	Activating the scene capture components with another overlapping actor. There are a couple of different ways to assign a 'triggering' actor. Player Controlled Pawn is set on default, but Specified Classes or Objects can add more then one actor to be the trigger.
Activation Tag	If Activation Tag is specified, the triggering actors must have the tag on them to activate the scene capture.
Filter From Classes	Map (array), add classes that may trigger the scene capture components. First, choose the actor class and then a component class inside it. The component will be the trigger and activate the scene capture when being overlapped.
Filter From Actors	Map (array), add actors that may trigger the scene capture components. First, choose the actor from the scene and then a component class inside it. The component will be the trigger and activate the scene capture when being overlapped.
Include Components	Include scene components when filtering objects or classes. Set variable to false or leave component class blank at Filter From Classes or Objects, to assign the actors scene root component as the trigger instead.
Manual Trigger	Trigger for the "Manual On / Off" activation method.
Manual Off Clear All*	Clear all render targets when the trigger is set to false, for the "Manual On / Off" method.
Player Pawn Capsule Only Trigger*	Only a capsule component inside the player pawn can trigger the 'Trailmarker' (Controlled methods only)

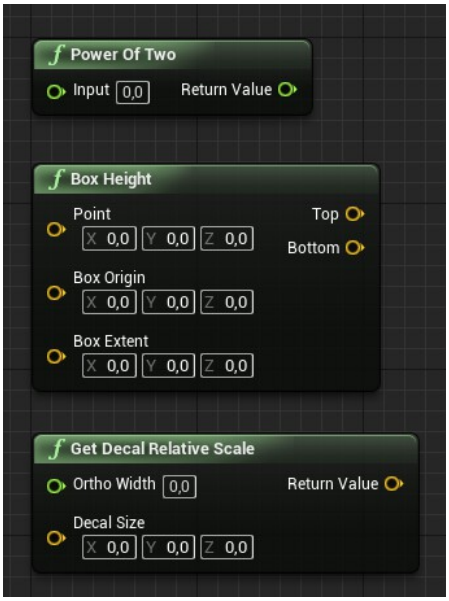
Settings|Find Nearest Actor

Actor Tag	If Actor Tag is specified, then only actors with that tag can be found.
Specified Actor Class	Find actor from a specific class (Optional)
Minimum Actor Size	Smallest actor allowed.
Maximum Actor Size	Optional, it's disabled when lower or equal to minimum actor size.
Ignore Actors Simulating Physics	Ignore any actor simulating physics, this is true on default.
Include Additional Actors	Include search result to additional actors. *When checking this box, the additional actor's map array will get flooded by every other actor this function has found. Unchecking this box, when the 'additional actors' have been added, will not delete them but make it possible for you to edit the map. It May be handy when wanting most of the actors to receive the projection, instead of manually filling them in one at a time to the additional actor's map array.*

** Hidden variables*

Toolkit a Blueprint Functions Library [Part I]

'TrailmarkerToolkit_BPL' is a library of functions that are to work with other blueprints then exclusive to the Trailmarker_BP. There are a few specific functions in this library that probably never would be useful in other blueprint applications and a few that might.

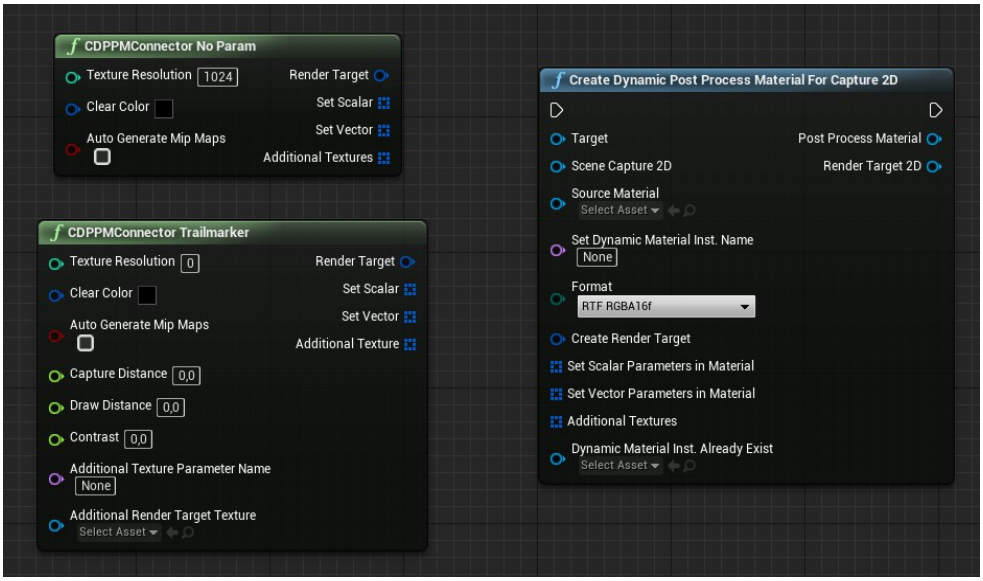


Math

Power of Two
This node will return the closest value of power of two from its input value.

Box Height
This node will output the top and bottom location based on the box origin. Point vector is setting location in X, and Y. Box origin and Box extent are by adding and subtracting find out where the top or bottom location is.

Get Decal Relative Scale
Find out what the relative scale should be from the Decal Size (static value) and Orthographic Width. (Decal Size x2 / Ortho Width)



Create Dynamic Post Process Material For Capture 2D

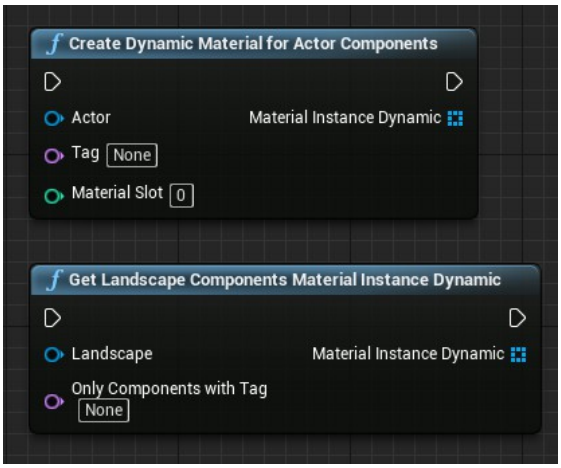
Create Dynamic Post Process Material For Capture 2D
This node is setting up a post-processing material instance dynamic for a Scene Capture 2D component. General information is required, such as source material, render target texture resolution, and color. The dynamic material instance can receive values to its parameters from the structure arrays. An important feature is the Dynamic Material Inst. Already Exist to avoid creating more material instances then necessary (Input a reference from its output here).

CDPPMConnector Trailmarker
Custom bridge to input values to the parameters included in the Trailmarker post process material. Used instead for making arrays.

CDPPMConnector No Param
Custom bridge that will exclude all parameters, If the source material doesn't have any parameters that should be set from the blueprint.

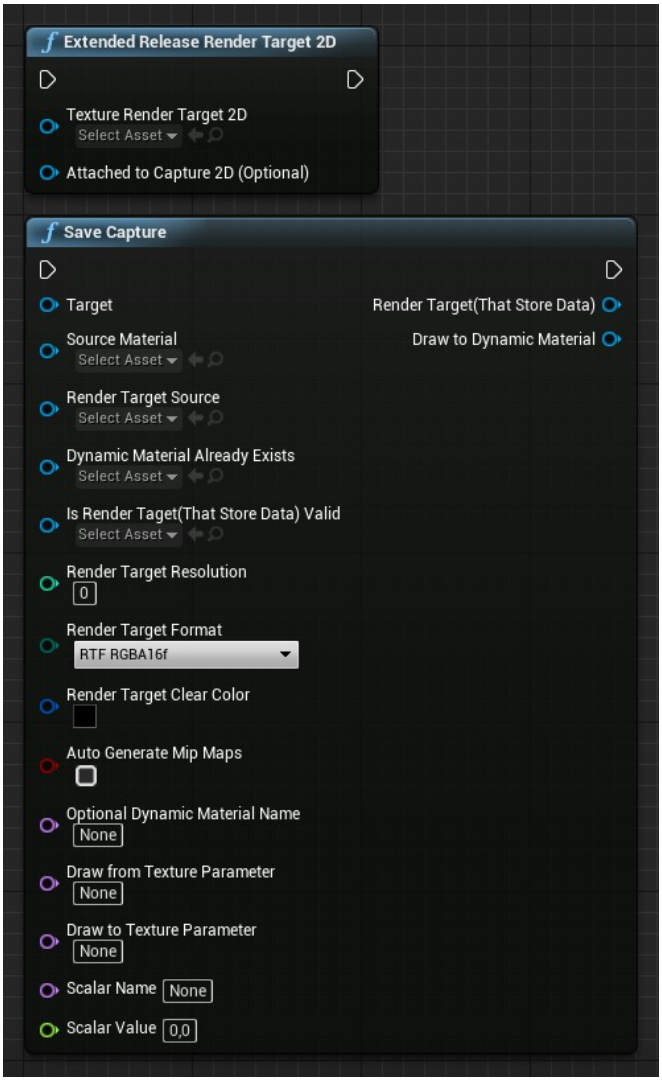
Toolkit a Blueprint Functions Library [Part II]

Material Instance Dynamic



Create Dynamic Material for Actor Components
This node creates a dynamic material for all or tagged static and skeletal mesh components inside the referred Actor (input).

Get Landscape Components Material Instance Dynamic
This node creates a dynamic material for all or tagged landscape components from the referred Landscape (input).

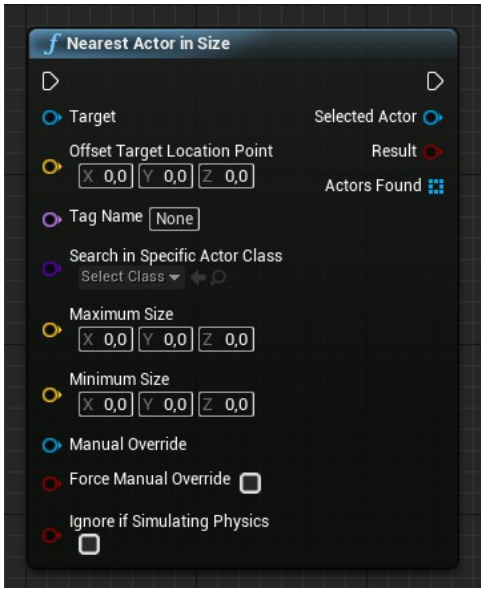


Scene Capture 2D and Nearest Actor in Size

Extended Release Render Target 2D
The node will release the render target 2d if valid. The optional feature Attached to Capture 2D, is setting the Scene Capture 2D 'set active' to false and clear its texture target.

Save Capture
This node is taking the current scene capture render-target and if it isn't already existing, creating a new dynamic material instance and render target to save the current marking on, like on a drawing canvas.

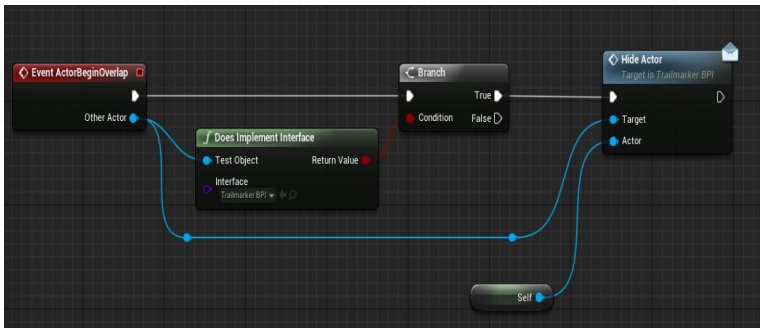
Nearest Actor in Size
Find the closest actor to target in a specific size. Size is measured by its bounding box.



Toolkit Remote Control a Blueprint Interface [Part I]

Implement an interface

Message a Trailmarker through its interface, from another blueprint to change how it will interact with any new actor at runtime.



Interface Functions

Hide Actor
Hide input actor reference from being captured by the Trailmarkers scene capture component.

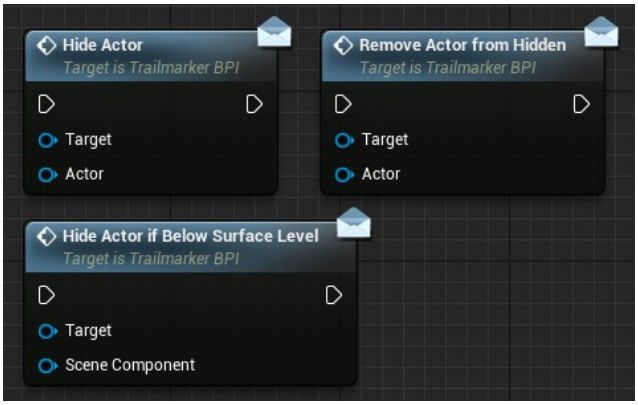
Hide Actor if Below Surface Level
Compare actor location to surface level, and hide the actor if it's below set height.
Require: Below Surface Level set to true.

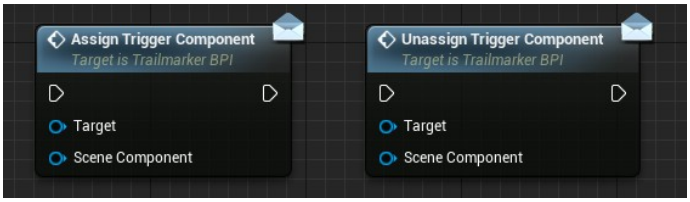
Remove Actor from Hidden
If actor are hidden from the scene capture component, make it visible again.

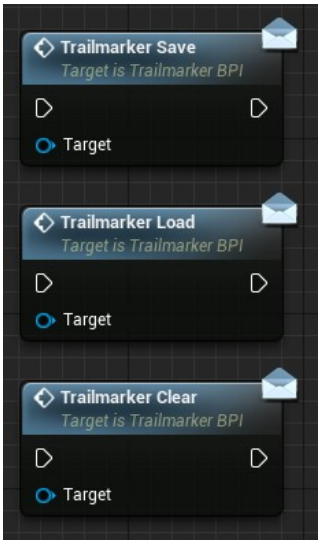
Assign Trigger Component
Assign a scene component inside an actor to trigger the Trailmarker, make it begin capture when this component are inside its volume.

Actor component have to be visible and, be able of generating overlap events.

Tip: Characters use their collision capsule to generate overlap events.







Trailmarker Save
Save markings on disk.

Trailmarker Load
Load markings from disk.

Trailmarker Clear
Clear markings on saved file at disk.

Note: The save, Load, and Clear interface doesn't require the Manage Render Target in settings to be set to something specific, like Save & Load. It will work regardless of the settings.

Saved and exported a render target to disk won't be deleted unless installing a 3rd party code plugin that can delete the file (see page 17).

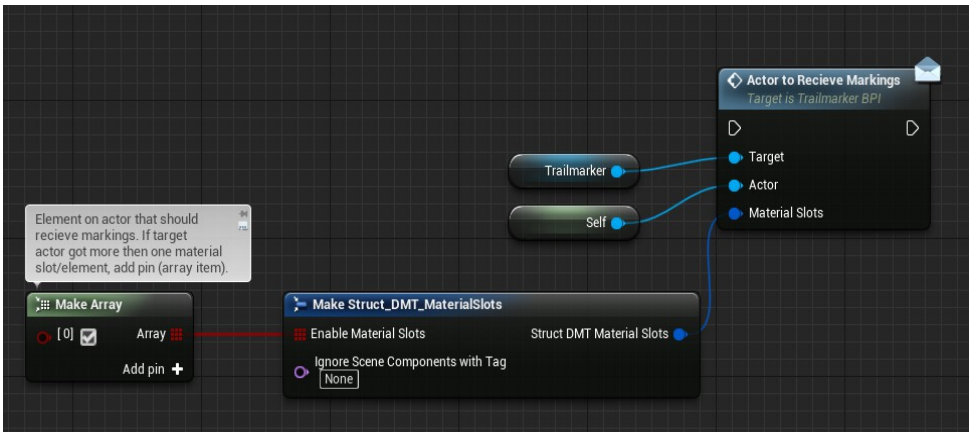
Toolkit Remote Control a Blueprint Interface [Part II]

Interface Functions

Actor to receive markings
Assign an actor* to Additional actors array and display markings on its components.

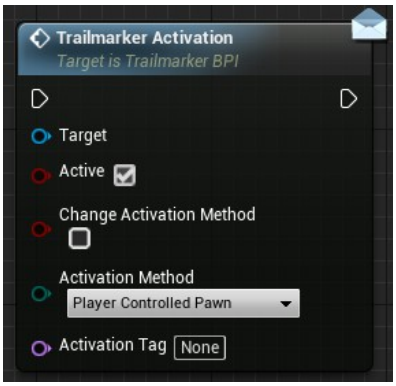
Material slots need at least one element assigned, make an array.

** Image (right) reference to self as the actor that should recieve markings.*



When actor(s) been added through 'actor to receive markings' nothing will happened until the Trailmarker implements them. 'Reload Trailmarker' will add new actors to relevant arrays and reload the Trailmarker.

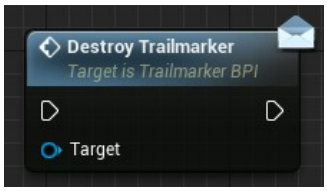
Note: There are no unrecieve markings, but can be done through casting.
Example found in SC_SpawnAdditionalActors commented as Despawn Ground Object.



Trailmarker Activation
Active: Turn a marker on or off

Change activation method:
Set the boolean to true, and the Trailmarker will set the new Activation Method to the input value of 'Activation Method' and an Activation Tag if specified.

Destroy Trailmarker
This will clear all markings and delete the Trailmarker.



Note: Destroying a Trailmarker with saved data will leave the exported file on the disk.

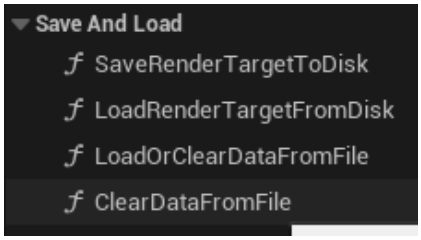
Tip: Mini Tutorial: How to add Delete File from a third party plugin on the next page.

Mini Tutorial: Delete Exported Save File (3rd Party)

Trailmarker can export saved render targets to file and load. But it cannot delete files from disk at runtime without any third-party plugins. Destroying a Trail marker with data exported to a file will leave the saved file on the disk. It's not a problem with a Trailmarker already placed in a level to have a savefile to modify. But it can be a problem with dynamically spawned Trailmarkers over many levels if all of them are saving data to disk.

With help of 3rd party plugins, we can delete the exported savefile before destroying the Trailmarker.

Have your 3rd party file manager plugin installed and activated in your project.



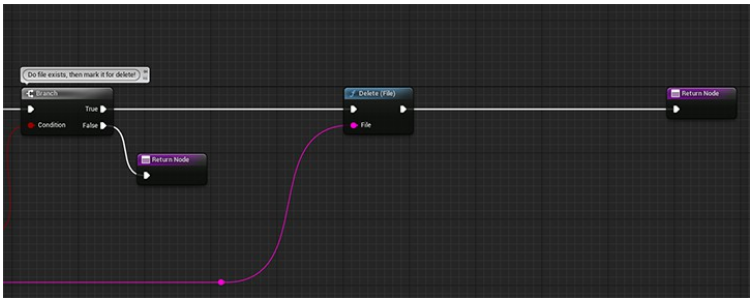
1. Open Trailmarker_BP

2. Browse functions in category Save And Load.

3. Duplicate ClearDataFromFile

4. Rename duplicate to ex. 'DeleteSaveFile' and open it.

5. Delete everything after the branch 'Do file exist'.
('Do file exists, then save a blank render target to clear')



6. Add 3rd party delete file blueprint node. If the file do exist. *(← image, right side).*

Drag the file path to the 'Delete File' input.

7. Go to Trailmarker_BPI blueprint interface and open DestroyTrailmarker function. Add boolean input ('DeleteSaveFile') and save.

8. Go back to Trailmarker_BP and open the Event Graph.
At the bottom in the red comment named Destroy Trailmarker.

Add branch between the delay and destroy actor.
Delay – Branch:True – DeleteSaveFile function – Destroy actor.
Delay – Branch:False – Destroy actor.

From Event DestroyTrailmarker boolean DeleteSaveFile connect it to the branch.

It should now delete the saved render target when message DestroyTrailmarker through its interface.

Trailmarker Content

Trailmarker

Trailmarker_BP – Blueprint
TrailmarkerSettings_BPC – Blueprint Component
TrailmarkerToolkit_BPL – Blueprint Function Library
MovingPlatform_BP – Blueprint (Example)
SpawnTrailmarker_BP – Blueprint (Example)

TrailmarkerActivation – Enumeration
TrailmarkerBounds – Enumeration
TrailmarkerComponent – Enumeration
TrailmarkerMemory – Enumeration
TrailmarkerProjections – Enumeration
TrailmarkerSaveDIR – Enumeration

Struct_DMT_MaterialSlots – Structure
Struct_DMT_RenderTarget – Structure
Struct_DMT_Scalar – Structure
Struct_DMT_Texture – Structure
Struct_DMT_Vector – Structure

M_TrailmarkerMaterialFunctionExample (9x Material Instances) – Material
M_TrailmarkerDecalExample – Material
M_TrailmarkerDraw – Material
M_TrailmarkerLoad – Material
M_TrailmarkerTerrainSLC – Material
PPM_Trailmarker – Material (Post Process Material)
PPM_TrailmarkerTerrain – Material (Post Process Material)

TrailmarkerDecal – Material Function
TrailmarkerWorldAligned – Material Function
Z+SurfaceMask – Material Function

Trailmarker_Icon – Texture
Masks_CubeshapedFalloff – Texture (3x Masks in RGB)
Masks_PuddleShapeFalloff – Texture (3x Masks in RGB)
Masks_SplatterShapeFalloff – Texture (3x Masks in RGB)

Demo Content

Important: UE4 content only for product demonstration.

Demo Landscape

Display Texts – Blueprint
M_BasicBlue – Material
M_TrailmarkerLandscape – Material
M_BlackMetal, M_Chrome – Materials
Grass-, Mud- and Stone_LayerInfo, – Landscape Layers
SM_DiscHolder – Static Mesh (352 vertices)
SM_Disc – Static Mesh (245 vertices)
SK_MovingPlatform – Skeletal Mesh (958 vertices)
*_Anim, *_PhysicsAsset – Animation, Physics Asset
SK_MovingPlatform_Skeleton – Skeleton
Island01_Splatmap – Texture

Demo ShowCase

BPInterface_Bounds – Blueprint Interface
BPInterface_MotionLink – Blueprint Interface
BPInterface_SaveLoadClear – Blueprint Interface
SC_BindBelowDepth – Blueprint
SC_BindButtonToTrailmarker – Blueprint
SC_DecalSaveLoadClearIcons – Blueprint
SC_DecalText – Blueprint
SC_DemoBall – Blueprint
SC_MovingPlatform – Blueprint
SC_SaveLoadAndClear – Blueprint
SC_SkeletalPlatform – Blueprint
SC_SpawnTrailmarker – Blueprint
SC_TextDisplay – Blueprint
SC_TrailmarkerBounds – Blueprint
M_DemoIconDecal – Material (12x Material Instances)
M_DisplayScreenBounds – Material
M_DisplayStandScreen – Material
M_SC_Landscape – Material (1x Material Instance)
M_Stripes – Material (2x Material Instances)
Brown_LayerInfo – Landscape Layer
Green_LayerInfo – Landscape Layer
Grey_LayerInfo – Landscape Layer
SM_DisplayScreen – Static Mesh (1890 vertices)
SM_DisplayDemoWall – Static Mesh (286 vertices)
SM_MultimaterialCube – Static Mesh (48 vertices)
SM_Staircase – Static Mesh (484 vertices)
SK_Platform_2 – Skeletal Mesh (5886 vertices)
SK_Platform_2_Anim – Animation
SK_Platform_2_PhysicsAsset – Physics Asset
SK_Platform_2_Skeleton – Skeleton
DisplayScreenInfo – Texture
icon_captureonmovement – Texture
icon_matvsdecal_inne_outer_bounds – Texture
icon_SaveLoadClear – Texture
icon_spawn_destroy_random – Texture
Stripes – Texture
TextDemoMotionProjection – Texture
… Demo content from updates aren't listed, release only.

UE4 Example Content

ThirdPersonCharacter – Blueprint
ThirdPersonGameMode - Blueprint
ThirdPerson_AnimBP – Animation Blueprint
SK_Mannequin – Skeletal Mesh (23297 vertices)
SK_Mannequin_PhysicsAsset – Physics Asset
UE4_Mannequin_Skeleton – Skeleton
ThirdPerson_IdleRun_2D – Blend Space
ThirdPerson_Idle – Animation
ThirdPersonJump_Start – Animation
ThirdPerson_Jump_End – Animation
ThirdPersonJump_Loop – Animation
ThirdPerson_Jump – Animation
ThirdPersonRun – Animation
ThirdPersonWalk – Animation
M_UE4Man_Body – Material
M_UE4Man_ChestLogo – Material
ML_GlossyBlack_Latex_UE4 – Material Function
ML_Plastic_Shiny_Beige – Material Function
ML_Plastic_Shiny_Beige_LOGO – Material Function
ML_SoftMetal_UE4 – Material Function
T_ML_Aluminum01 – Texture
T_ML_Aluminum01_N – Texture
T_ML_Rubber_Blue_01_D – Texture
T_ML_Rubber_Blue_01_N – Texture
UE4Man_Logo_N – Texture
UE4_LOGO_CARD – Texture
UE4_Mannequin__normals – Texture
UE4_Mannequin_MAT_MASKA – Texture

UE4 Demo Room Content

BPInterface_Button – Blueprint Interface
BP_DemoDisplay – Blueprint
BP_DemoRoom – Blueprint
BP_DemoRoom_Enum – Enumeration
BP_DemoTrigger – Blueprint
DemoRoomStruct – Structure
DefaultTextMaterialTranslucent – Material
DefaultTextMaterialTranslucentUnlit – Material
M_Button – Material (1x Material Instance)
M_Button_Chrome – Material
M_DemoRoomTiles – Material (1x Material Instance)
M_DemoWall – Material (6x Material Instances)
M_DemoWallPattern – Material (3x Material Instances)
M_Glass – Material (1x Material Instance)
M_LightTube – Material (1x Material Instance)
SM_Base – Static Mesh (6765 vertices)
SM_Button – Static Mesh (1515 vertices)
SM_DemoDivider_1 – Static Mesh (1164 vertices)
SM_DemoRoomBackWall – Static Mesh (66 vertices)
SM_DemoRoomClamp – Static Mesh (3419vertices)
SM_DemoRoomLTrim – Static Mesh (44 vertices)
SM_DemoRoomTrim – Static Mesh (5530 vertices)
SM_DemoRoomU – Static Mesh (1086 vertices)
SM_NamePlate – Static Mesh (216 vertices)
SM_URoom_Wall – Static Mesh (4 vertices)
SM_URoom_Wall2 – Static Mesh (4 vertices)
T_Pattern_M – Texture
T_RoomTiles_M – Texture
T_RoomTiles_N – Texture
T_UnrealEngineDecal – Texture

Update Notes

Release version 1.00.3
12/20/2021

Update 1 version 1.00.4
06/15/2022

- # Support for procedural meshes.
  It's not added to default classes in find nearest actor, but can be assigned with 'Override Main Actor'.

- # Include children on overlapping actor when using On Begin Overlap Disable Receives Decals and
  On End Overlap Enable Receives Decals. The function can be disabled from set the hidden variable
  bIncludeChildrenOnReceivesDecals to false.

- # Fixed: When player controller was changed at runtime the trigger reference returned null / accessed none.

Update 2 version 1.00.5
01/16/2023

- # Support for landscape streaming proxy.
  Included in default classes in find nearest actor.

- # Trailmarker Blueprint Interface added.
  Control the Trailmarker from other actors with interface functions.

- # Fixed: Destroy actor with overlapping components inside the volume resulted in null / accessed none error.

- # Fixed: Bounds set to Manual in settings (Settings|Capture) wasn't working correctly with offset.
  Capture Height Offset and Decals height offset are now working on bounds type manual.

  Note: Decal volume x and y are matching the orthographic width.

- # Showcase demo room updated to include interface functions and a few general improvements.

Version Notes

Unreal Engine 5.2
LogPackageName: Warning: DoesPackageExist called on PackageName that will always return false. Reason: Input '' was empty.
Trailmarker will create and remove components at runtime and Unreal may add this warning for each time it does that.
That's the “Unsaved” category in the Outliner. Right click the asterix in the outliner bar and uncheck “unsaved.”
Thanks popomark for solving this (Read the original comment at UE Forums) - *Note written 230703*

Contacts

Mail: support@morselbyte.com

Website: https://www.morselbyte.com