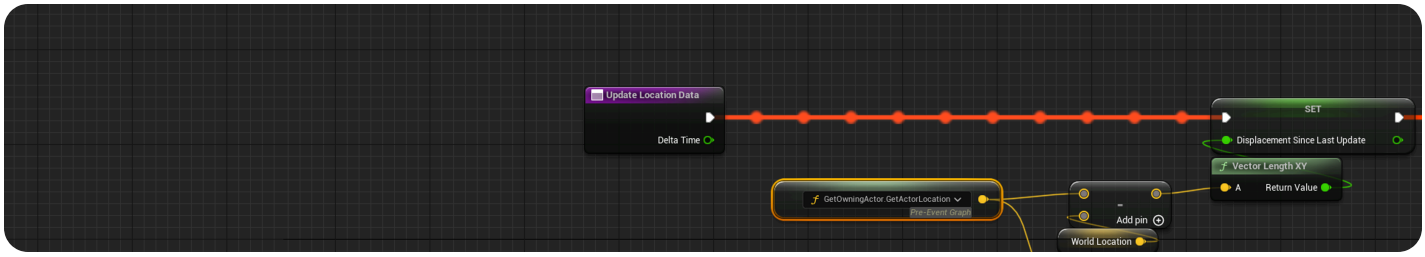
Property Access

Using Property Access Node Functions.



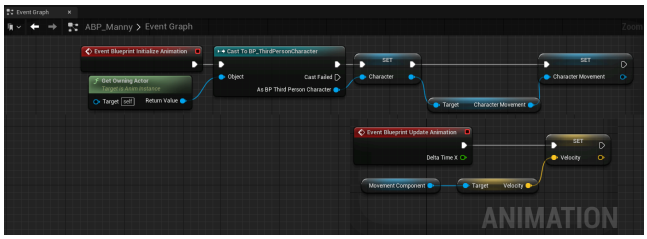
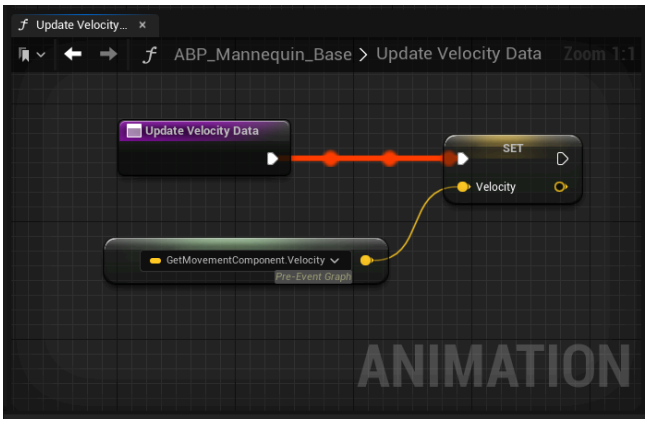
Property Access allows you to access components and variables that are only accessible on the **Game Thread** anywhere within an Animation Blueprint. Calling a component or variable using Property Access takes a snapshot of its data, which you can use to drive function logic in the Anim Graph, Animation Node Function, a Blueprint Thread Safe Function, or in the Event Graph. Using Property Access reduces the amount of work your project must evaluate on the **Game Thread**, and also reduces the amount of manual blueprint scripting you must perform, leading to increased game and workflow performance.

You can use the following documentation to learn more about using Property Access nodes, pins, and functions that you can use to optimize your project's animation system.

Overview

When creating logic in an Animation Blueprint, you will typically use the Event Graph to set variables and interface with other Blueprints and Components. The Event Graph is always evaluated on your project's **Game Thread**.

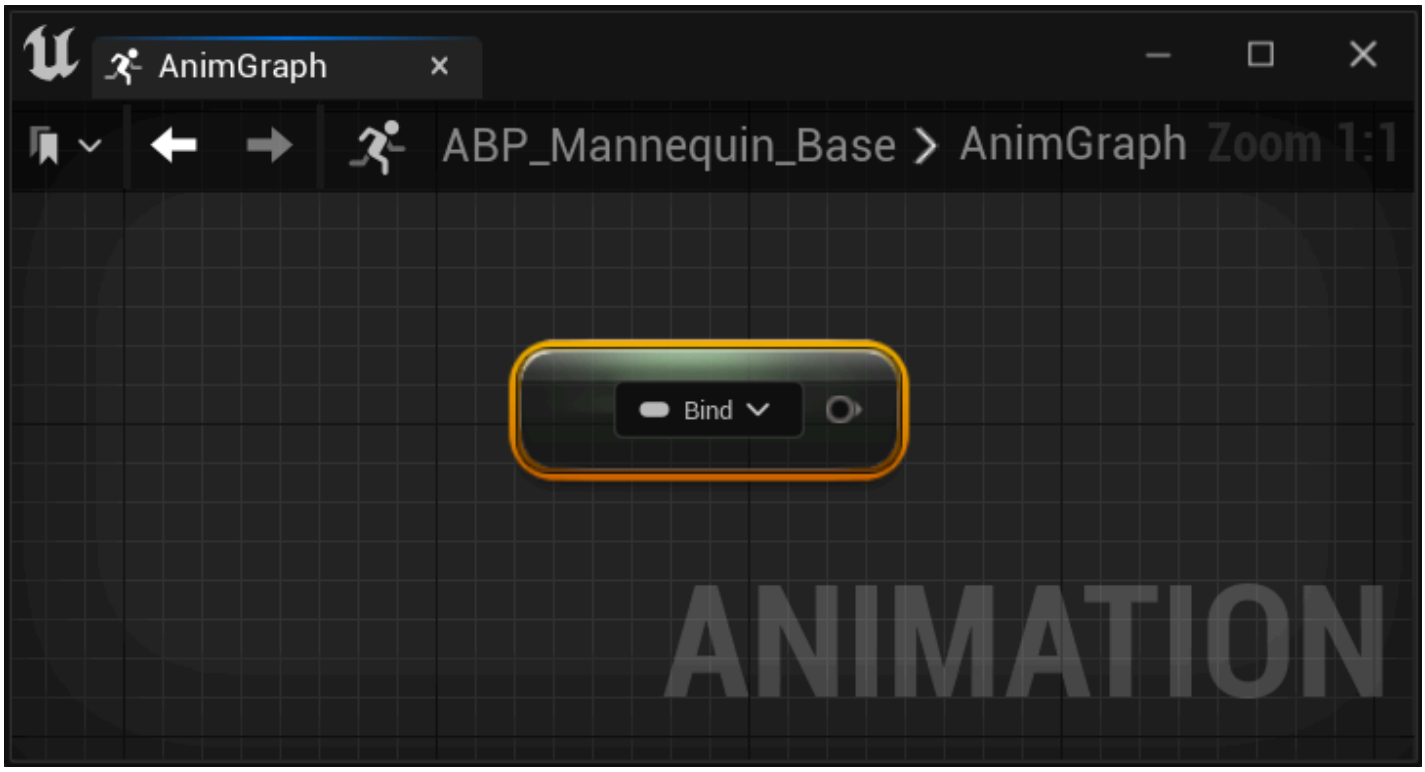
To increase your animation system's performance, you can run **Event Graph** functions on the same **Worker Thread** the Animation Graph is updating on, using **Property Access Functions**. For example, you can use Property Access to directly access the character's Velocity without the need to cast to the character blueprint and create a function node to extract the velocity vector.

Without Property Access	With Property Access
	

Property Access Functions

Property Access works by evaluating a function graph to take a snapshot of a Game Thread variable's or component's data, and converting that data to be Thread Safe compatible. When creating a Property Access node, or calling a Property Access function on an Animation Blueprint Node pin, you can select from a number of pre-generated functions that access your project's base components and variables.

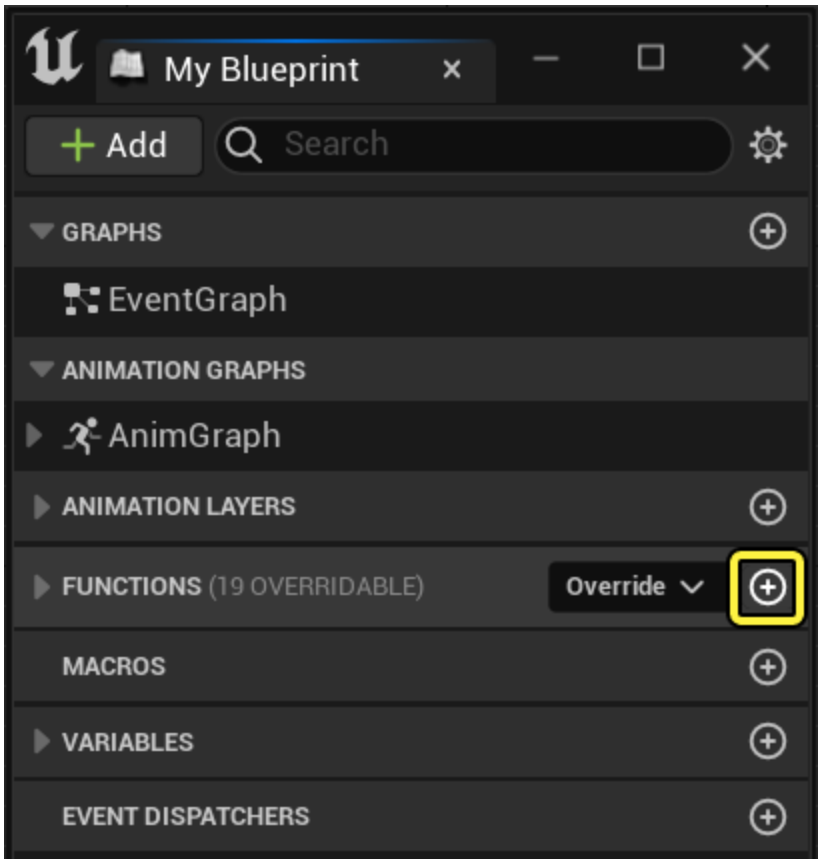
These functions can provide variables and components such as a character's velocity, movement direction, or a movement component reference, that you can use to drive animation pose selection or blending in the AnimGraph.



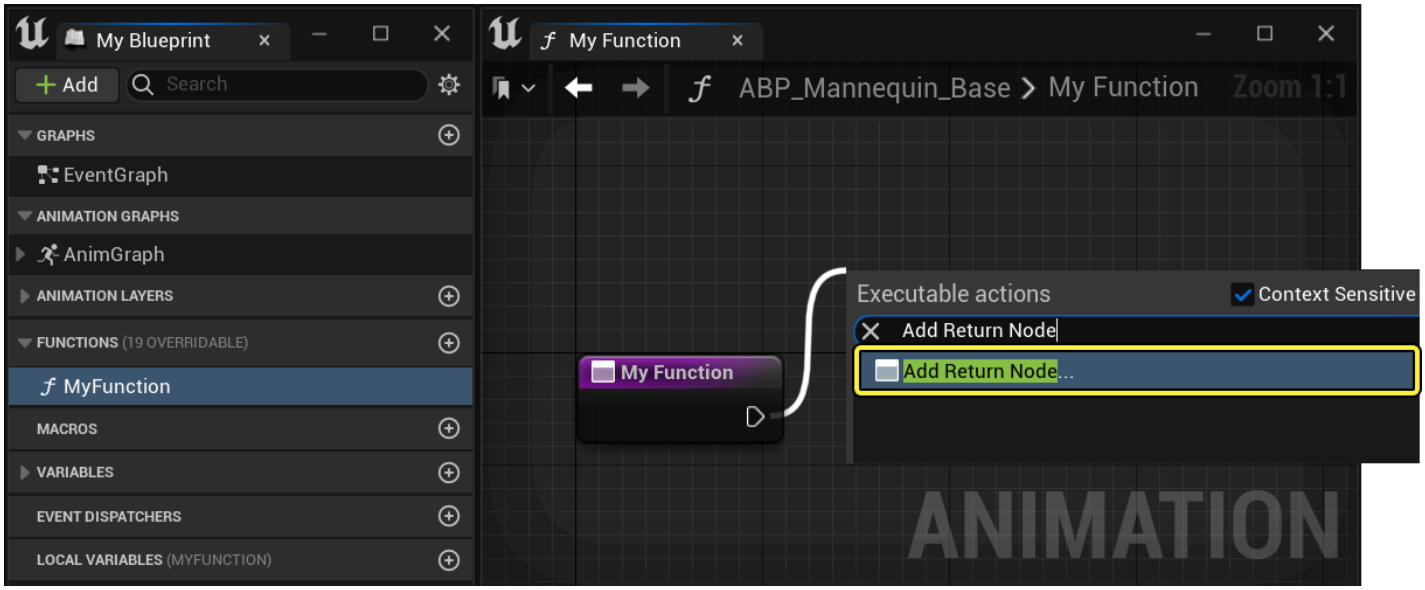
For more information about binding Property Access nodes and pins, see the Property Access Bindings section.

Custom Property Access Functions

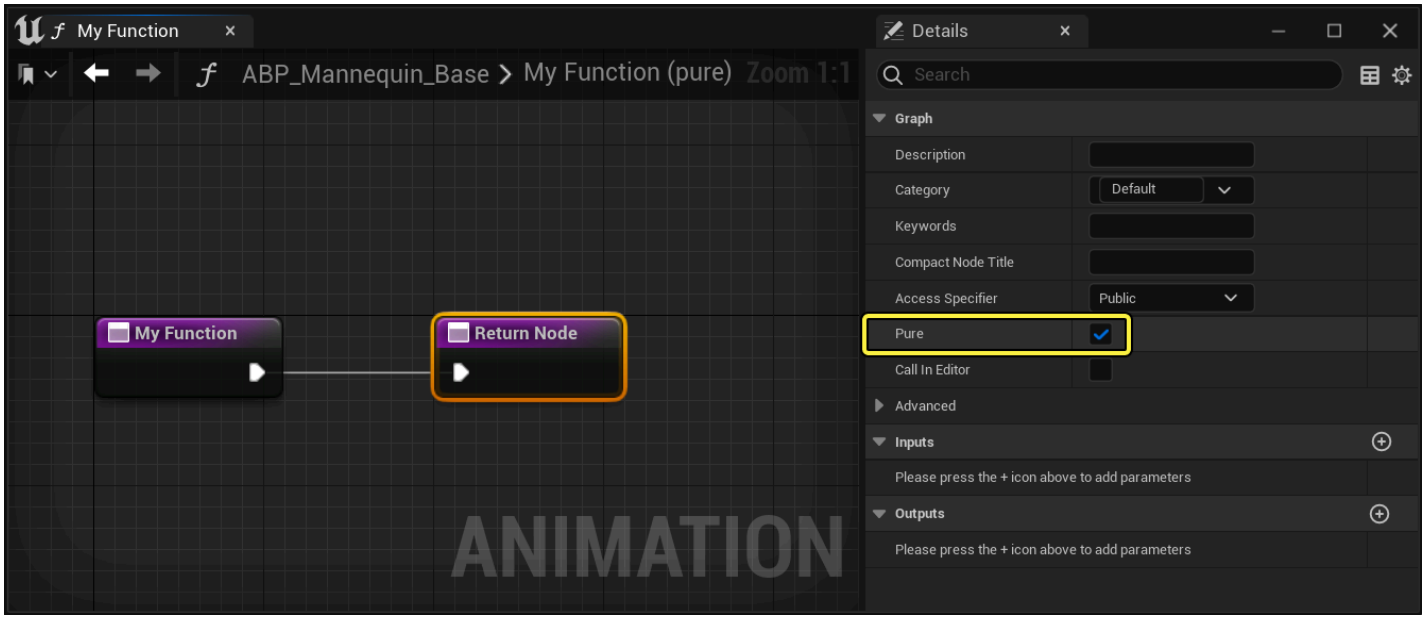
You can also create custom Property Access Functions in order to output more specific or unique values. To create a custom Property Access function, start by adding a new function to the graph, by selecting **Add** (**+**) in the **My Blueprint** panel.



Create a Return Node, and connect it to the Function Node.

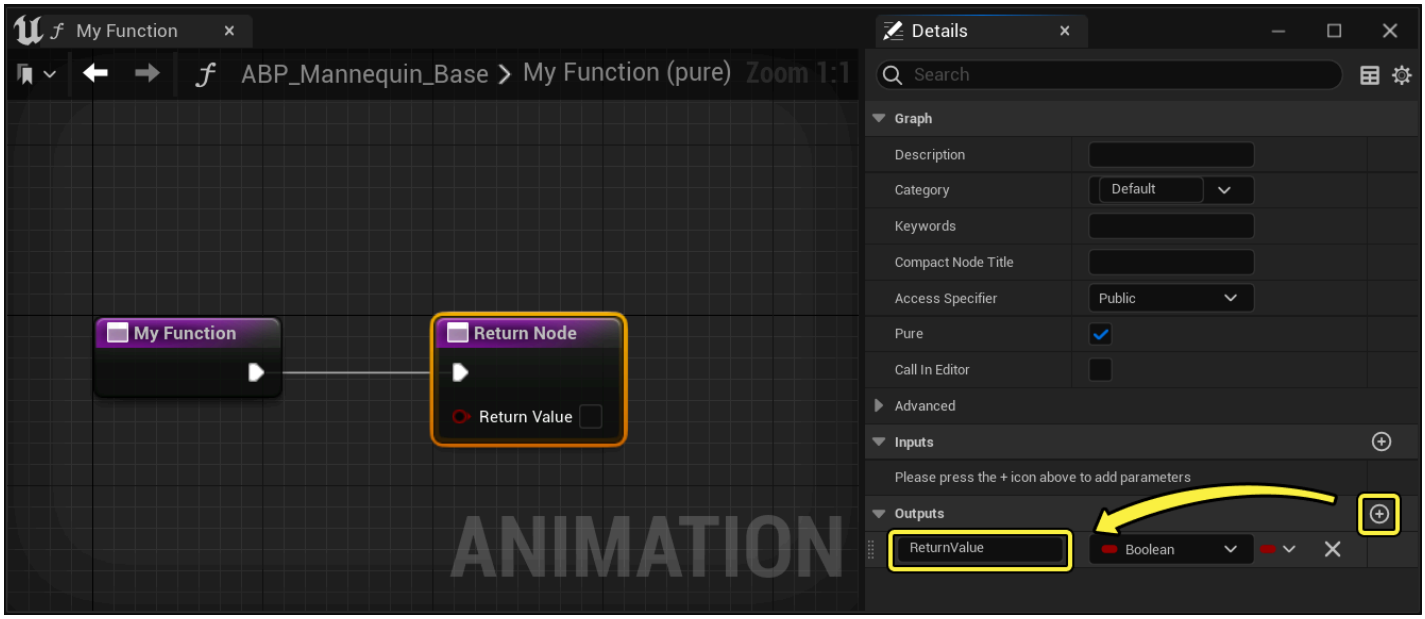


Select the **Return Node** and enable the **Pure** property in the **Details** Panel.



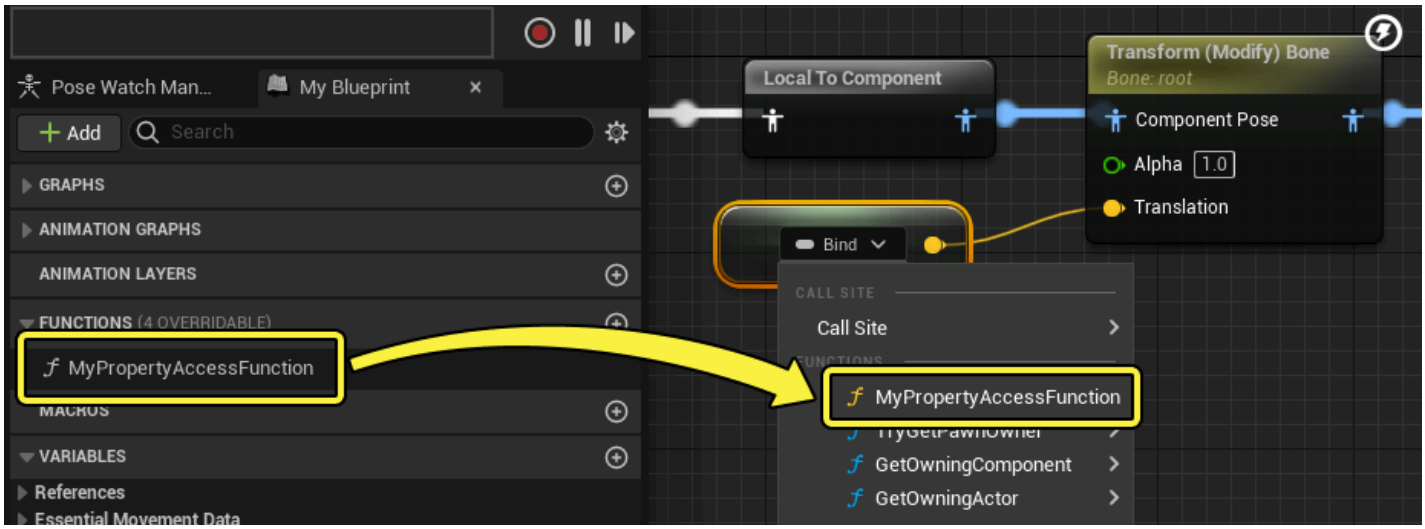
Add a new Output value using **Add** (**+**) and name the output, `ReturnValue`.

> The Output must be named `ReturnValue` exactly. Any other name will result in the Property Access Function being inaccessible from a Property Access node or pin binding.



You can now build any logic to set the Return Node's Output pin, which will then become the output of any Property Access node or pin bound to your function.

You can now bind the Property Access Function to any Property Access node or pin by selecting the function using the Bind drop down-menu. Property Access Functions are color-coded depending on the data type of the Output.



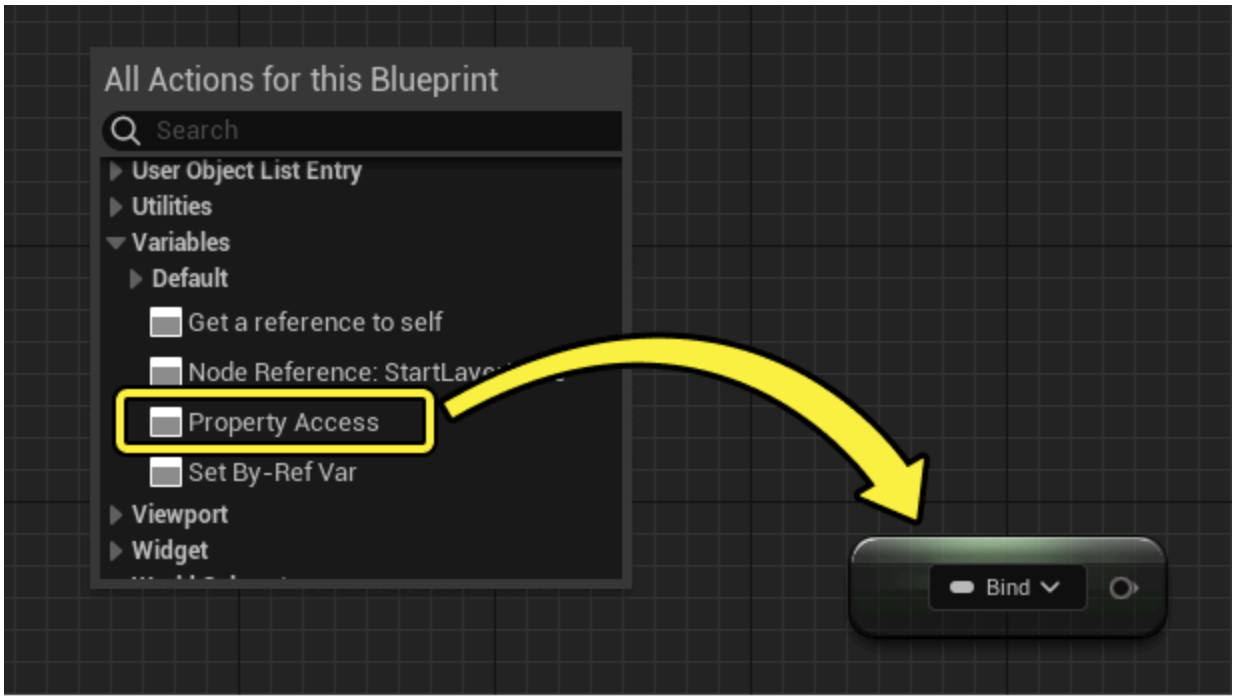
For optimized performance, we recommend you enable the Thread Safe property for Property Access Functions. After enabling the function as thread safe, you will need to make all of your variable and component references using Property Access as well. By making custom Property Access functions Thread Safe, you can significantly increase your animation system's performance.

Property Access Bindings

You can access default and custom Property Access Functions anywhere in an Animation Blueprint using either a Property Access node, or as a pin binding on an existing Animation Blueprint node.

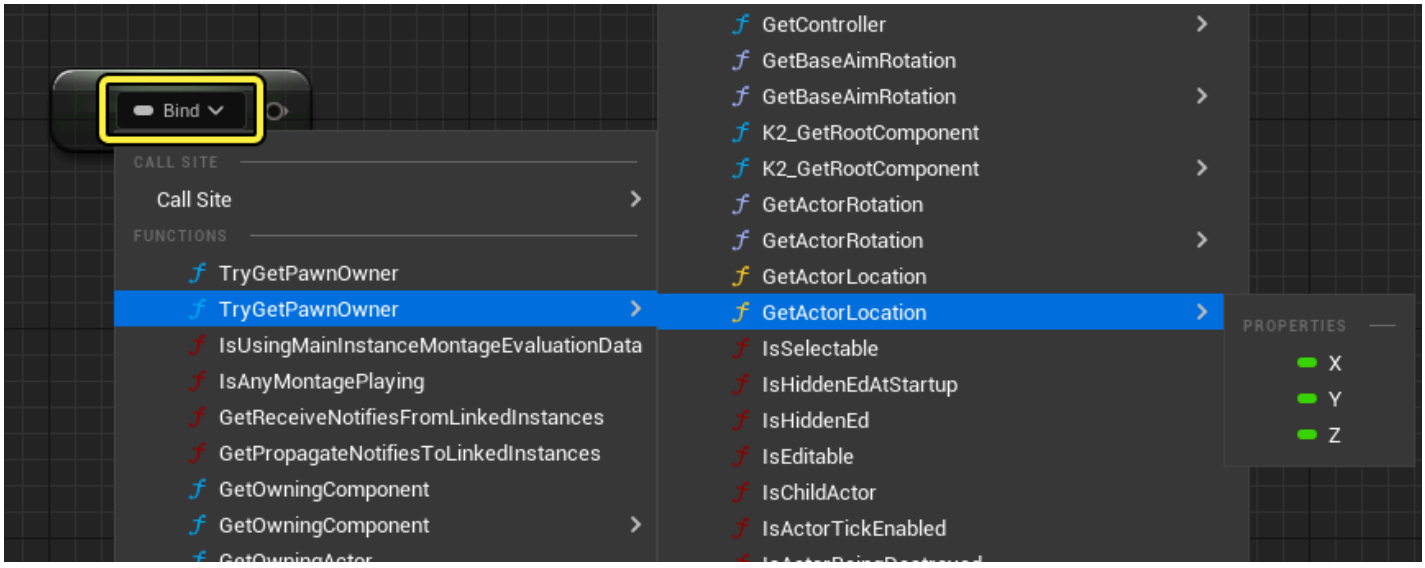
As a Node

You can create a Property Access node anywhere in an Animation Blueprint. To create a Property Access node, right-click in the **AnimGraph** and select **Property Access** from the **Variables** category.

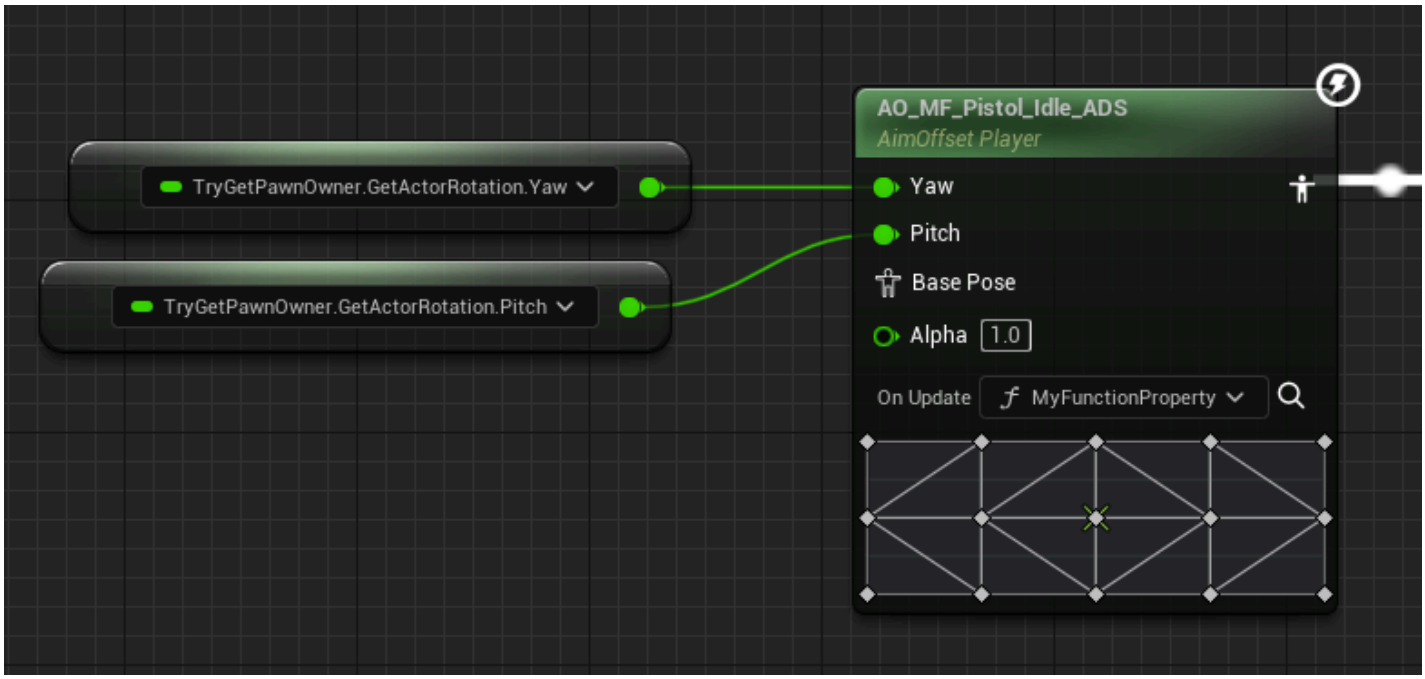


After creating a Property Access node, you will need to bind it to a **Function** to reference the component or variable you want to access. You can also bind custom Property Access function graphs, if they meet the necessary requirements. To bind a function to a Property Access node, select a Function from the **Bind** drop-down menu.

Functions are organized by hierarchy, with parent component functions containing child functions that correspond to their child properties. For example, you can navigate beyond a parent Function, to isolate a more specific property that it contains. In this example, a Get property path is created from **TryGetPawnOwner**, to **GetActorLocation**, to a specific axis float value (**X**, **Y**, or **Z**).

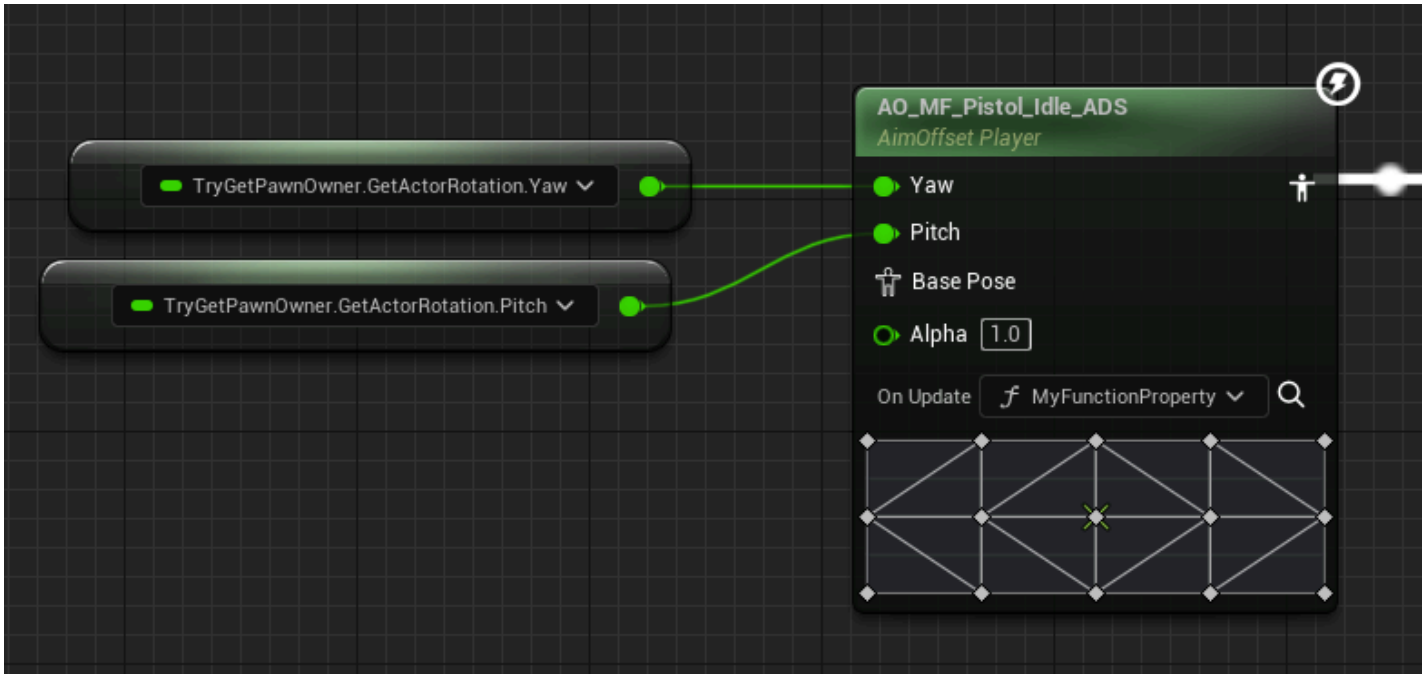


Once you bind a Property Access node, you can use the node to access the component or variable in your graph, to drive a node's logic.



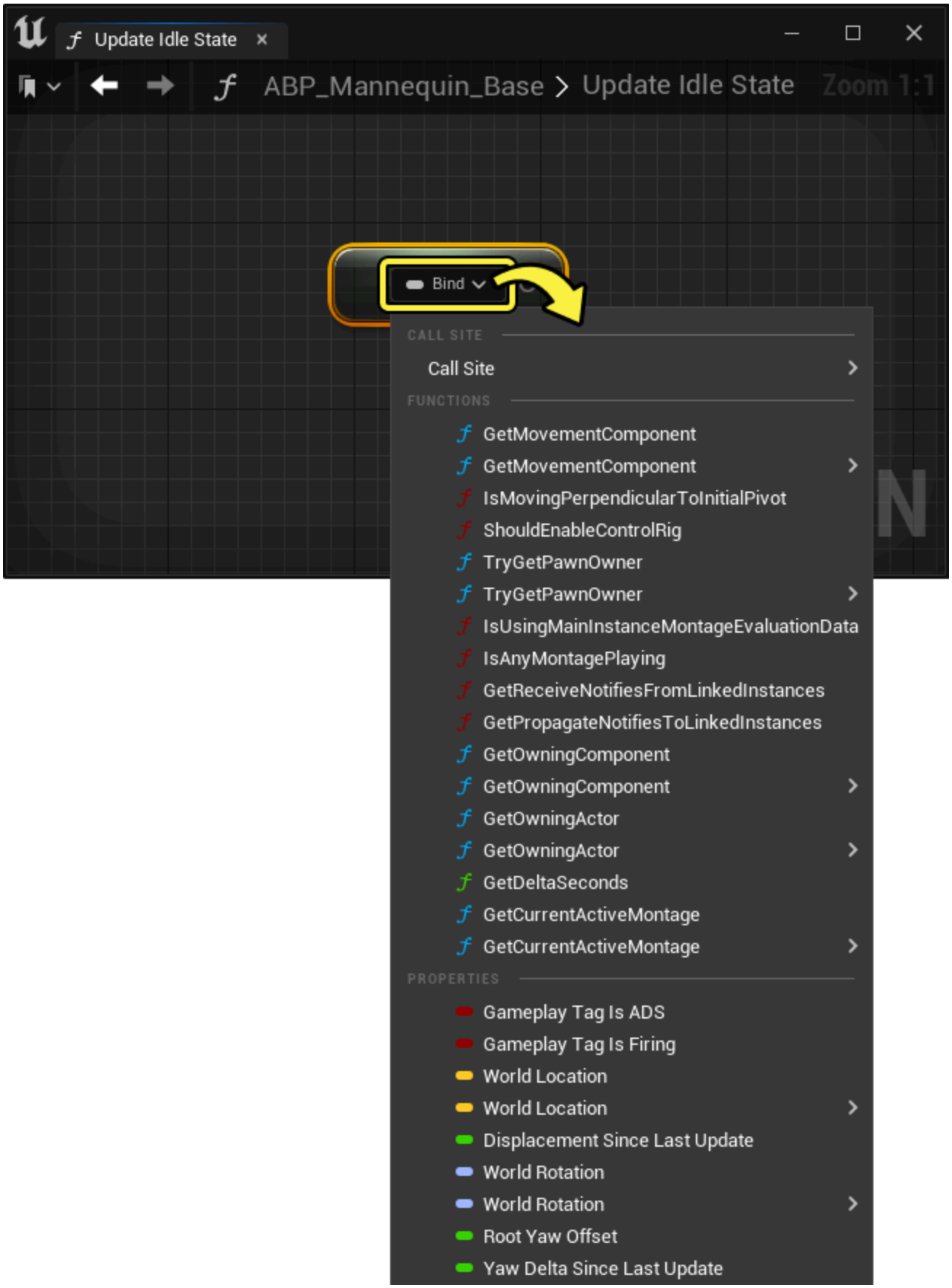
As a Pin

You can also bind Property Access Functions directly to some AnimNode pins. To bind a Property Access Function to an AnimNode pin, select the node in the graph and navigate in the node's Details panel to the Input pin binding settings. Select **Function** from the Pin drop-down menu to dynamically drive the pins value.



Property Access Settings

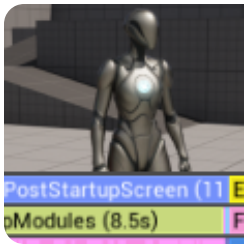
The Property Access context-menu contains the following selections and properties:


Update Idle State
ABP_Mannequin_Base > Update Idle State
Zoom 1:1
Bind
CALL SITE
Call Site
FUNCTIONS
GetMovementComponent
GetMovementComponent
IsMovingPerpendicularToInitialPivot
ShouldEnableControlRig
TryGetPawnOwner
TryGetPawnOwner
IsUsingMainInstanceMontageEvaluationData
IsAnyMontagePlaying
GetReceiveNotifiesFromLinkedInstances
GetPropagateNotifiesToLinkedInstances
GetOwningComponent
GetOwningComponent
GetOwningActor
GetOwningActor
GetDeltaSeconds
GetCurrentActiveMontage
GetCurrentActiveMontage
PROPERTIES
Gameplay Tag Is ADS
Gameplay Tag Is Firing
World Location
World Location
Displacement Since Last Update
World Rotation
World Rotation
Root Yaw Offset
Yaw Delta Since Last Update

Name	Description
Call Site	The **Call Site** controls which thread to execute the bound Property Access function on. You can select the following options: • **Automatic**: Automatically determines the **call site** for the bound Property Access Function based on context and thread safety. You should leave this as the selection in most cases. • **Thread Safe**: Evaluates the bound Property Access Function on the **Worker Thread**. • **Game Thread** (**Pre-Event Graph**): Evaluates the bound Property Access Function on the **Game Thread**, before the **Event Graph** is evaluated. • **Game Thread** (**Post-Event Graph**): Evaluates the bound Property Access Function on the **Game Thread**, after the **Event Graph** is evaluated. • **Worker Thread** (**Pre-Event Graph**): Evaluates the bound Property Access Function on the next available **Worker Thread**, before the Event Graph is evaluated. • **Worker Thread** (**Post-Event Graph): Evaluates the bound Property Access function on the next available** Worker Thread**, after the** Event Graph** is evaluated.
Functions	Select a Property Access Function to bind to the node or pin.
Properties	Select a Variable or Component Reference contained within your Animation Blueprint, that you can bind to the Property Access node or pin.

Additional Resources

For more information about using Blueprint Thread Safe Functions and Property Access to optimize your animation system, see the Animation Optimization documentation.



Animation Optimization

Use a variety of methods and techniques to optimize Animation Blueprint's performance and stability.

For a workflow example of using Blueprint Thread Safe Functions and Property Access to get Animation Variables, see the How to Get Animation Variables documentation.

How to Get Animation Variables

How to get Animation Variables to animate characters in the Animation Blueprint Event Graph and Thread Safe Functions.

For a workflow example of optimizing an Unreal Engine project to use Thread Safe functions, see the Adapting Lyra's Animation System blog post.