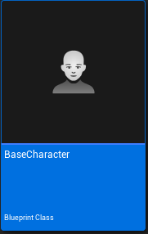
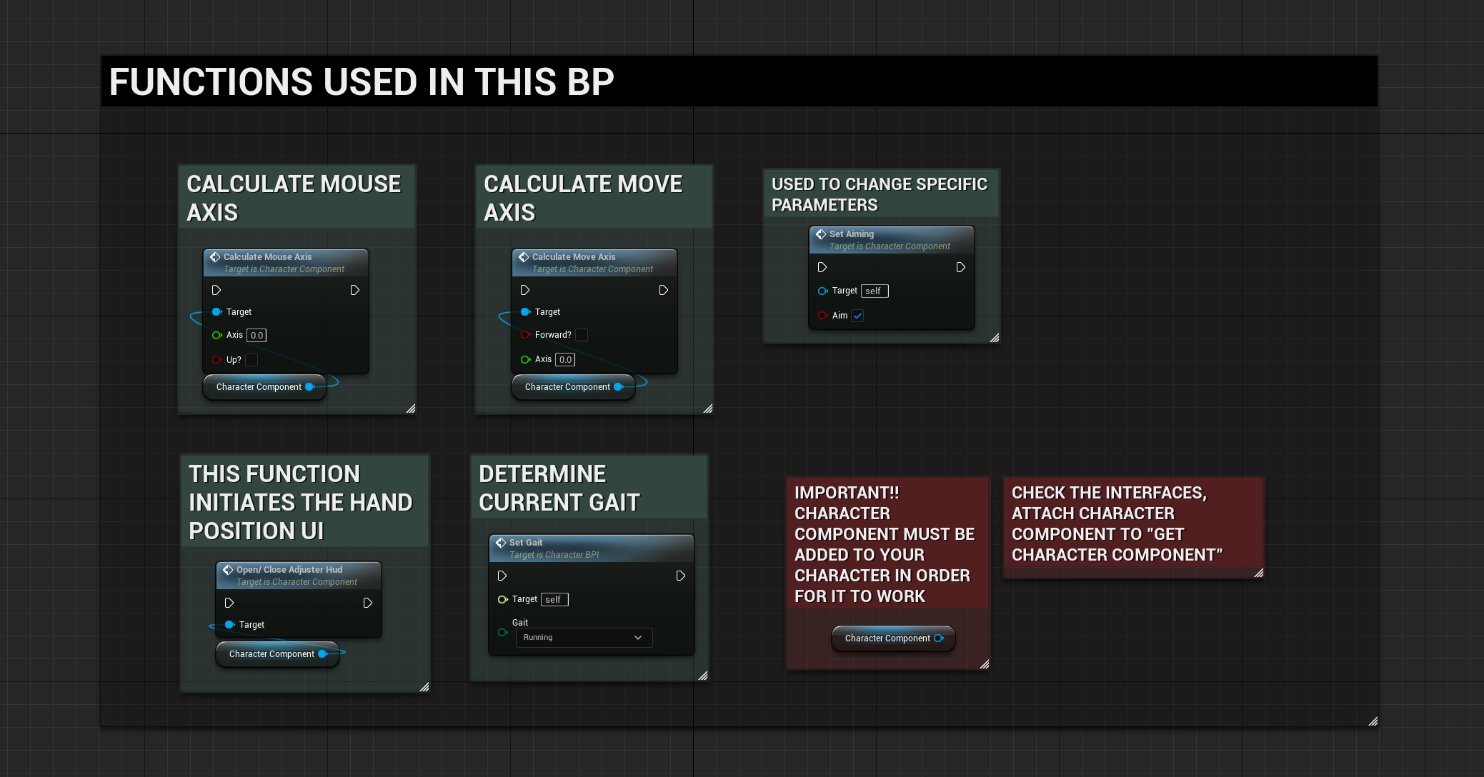
Procedural Character Component

Example content overview

***Base character***

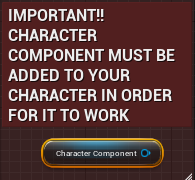
In the example content there are a few things that go into making the procedural animations work. First let’s look over our “base character” blueprint.



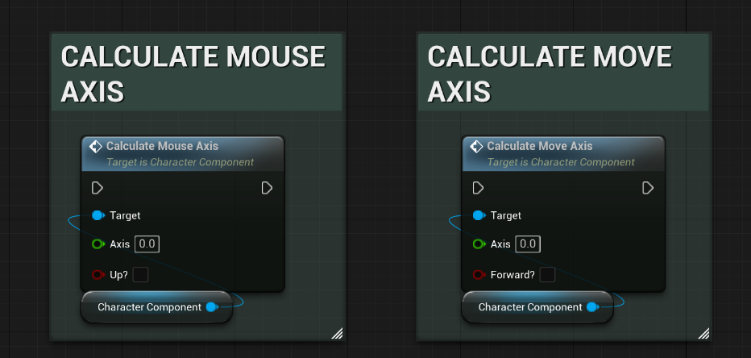
****

These are all the functions that are being utilized within the base character. Let’s go over each one and specify their functionalities.

*Character component*

******

Adding the component should be one of the first things that needs to be added to the character. This component has all the functions that we need to make our procedural adjustments work.

*Calculating axis*

These two nodes are self-explanatory, we use them to calculate the mouse and key inputs. The variables that are set in these functions are used to calculate our character sway. The way we use these nodes is by attaching them to our input events.

*Set Aiming*

This function is just a simple Boolean and can be extended to your liking. The way it is used in the component is by alternating our spring and sway values to a smaller value. For example, if you are aiming, the character will decrease the sway and make the wobble effect stiffer.

A screenshot of a computer

Description automatically generated with medium confidence

*Open Close Adjuster Hud*

This function simply opens and closes the main HUD where the parameters can be altered. You can use any key input to open and close the UI.

****

*Set Gait*

In order for our values to change dynamically we must set this to each specific action. Each gait has its own Walk Speeds, it is made simple so the developer (you) can extend this by having a weight system for example.

**A screenshot of a computer

Description automatically generated with medium confidence**

*Character BPI* 1

The Character BPI needs to be under the implemented interfaces. The reason we are utilizing it in our character is to retrieve the character component from a character parameter. Additionally, we are using it to determine the current gait the character is in.

**A picture containing text, screenshot, font

Description automatically generated**

***Third Person Character***

This blueprint is the character that should get spawned when playing in the editor. The logic within this blueprint was kept simple by only using these two functions below.

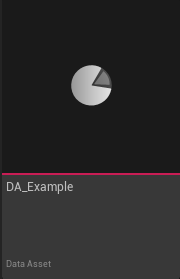
A screenshot of a computer screen

Description automatically generated with low confidence

***Update Held Object****:*This function gets all parameters that are set within a DA (Data Asset). The function is only activated if a held object is valid, if it is not valid the function must have an empty DA. An example of when this function should be active would be when a character is switching weapons.

**Data Asset**

The data asset holds multiple structs, the structs contain parameters that are used to dynamically change the location/rotation of the object in hand. In addition, the parameters that are used for the wobble effect are also included in the DA. The reason a DA is being used is in the case the character has different objects to be held. The DA can be set and called by a blueprint containing the specified DA. This gives each held object their own unique adjustments.

****A screen shot of a computer

Description automatically generated with low confidence

In this picture is the inside of the DA, all values here can be set using the adjuster HUD. However, if the values need some quick adjustments, they can also be altered within the DA.

***Hand IK*:** The hand IK function is used to allow the character to always maintain the left hand on the held object. The way this works is by setting a socket named “IK” on an object that the character will hold. When this function is active it calculates where the transform of the socket would be and adjusts the left hand to that location automatically. The object mesh your character is holding should be connected to the “HELD ITEM” pin.

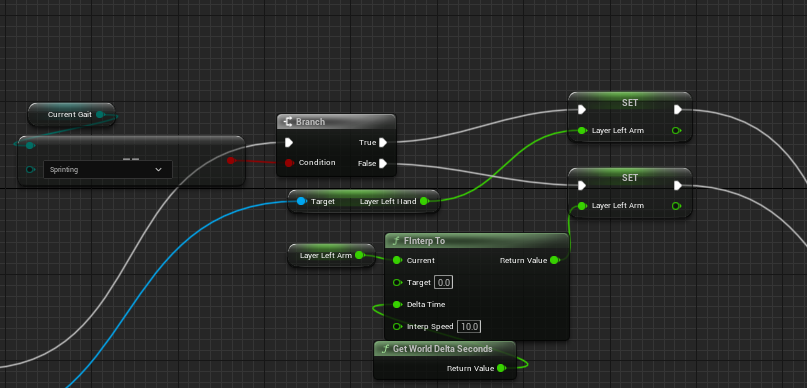


***Character Component***

*To get access to the documentation needed for the component please join the discord. There you will receive extensive docs on how the character component functions. Please give your order ID to “Crypto” and you will gain access to the support channels regarding the asset.*

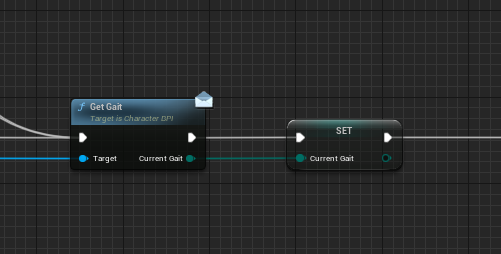
Discord Link: <https://discord.gg/phantom-814427844222517268>

***Main Animation Graph***



This line of code is used to determine if the character will use their left hand on sprint. It can be altered using the “Adjuster Hud” or by simply changing the float value to 1.

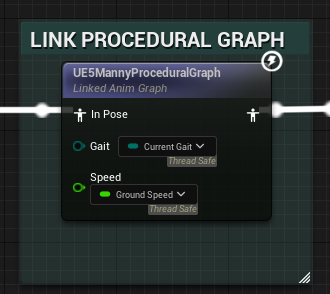
*Get Gait*

**

The get gait message is called via the character. This sets the character gait in the main animgraph and is utilized in the procedural graph.

*Linked Graph*

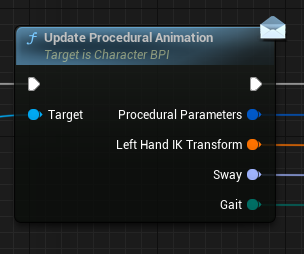
This is the most important node that must be attached to make the procedural animations work. As you can see, we are using the properties located in the main animgraph in our linked graph. This is to change those values via the main animgraph values.

**

***Procedural Animation Graph***

Let’s look at the main function that drives the procedural animations.

*Update Procedural Animation*



This function is in the Character BPI. All Values are set within the component and are called using this function. Looking at the picture below we have broken the “Procedural Parameters” struct. You will notice that we have floats and transforms. All floats and transforms will be set and called using the “Procedural Parameters” struct within the procedural character animation graph.

A screenshot of a computer program

Description automatically generated with medium confidence

***HUD***

*Adjuster Hud*

A screenshot of a video game

Description automatically generatedThe adjuster hud is the main logic that carries this entire system. It is made easy to use and can be extended easily as well! (information used in interface located in discord support channel)

*Wobble affect*



This section is able to change the amount of the wobble the arms have. You can select the spring to modify such as Roll, Pitch, Yaw, Xloc, YLoc, and Zloc from the tab. All action sections such as walk, run, sprint ect. Can be modified by adjusting the values.

*Hand offset*

**

This section works in a very similar fashion. The gait that the dev would like to modify can be selected by the tab. Once the tab is changed the values displayed should change automatically. Here the dev can change the transform of the arms allowing the dev to choose the pose they would like for the held object.