**How-to First Person Melee System in Unreal Engine 5**

Contents

[**Comments & color coding:** 2](#_Toc181370162)

[1. Naming Conventions 3](#_Toc181370163)

[**Changelog:** 4](#_Toc181370164)

[18/01/2024 4](#_Toc181370165)

[**Downloads and Set-up.** 4](#_Toc181370166)

[**Download Unreal Engine** 4](#_Toc181370167)

[**Download and migrate the project.** 4](#_Toc181370168)

[**Gym** 4](#_Toc181370169)

[**DemoLevel** 4](#_Toc181370170)

[**Video Content:** 5](#_Toc181370171)

[**Creating Content** 5](#_Toc181370172)

[How to set up a level 5](#_Toc181370173)

[1. How to set up an actor. 6](#_Toc181370174)

[Setting up the Nodes. 9](#_Toc181370175)

[2. Creating a weapon 15](#_Toc181370176)

[3. Setting up Animations. 19](#_Toc181370177)

[4. Misc 21](#_Toc181370178)

# **Comments & color coding:**

This image provides some context on the different colors used for all the code, it’ll help you jump through it more quickly and find the areas you’re interested in.

A screenshot of a computer screen

Description automatically generated

## Naming Conventions

For the naming conventions I tried to stick to the following rules:

* **Audio**
  + **Sound Cue:** SC\_\*Name\*
  + **Sound Attenuation:** SA\_\*Name\*
* **Blueprints:**
  + **Parent**: BP\_\*Name\*
  + **Child**: BPC\_\*Name\*
  + **GameMode**: GM\_\*Name\*
  + **HUD**: HUD\_\*Name\*
  + **Game Instance**: GI\_\*Name\*
  + **Blackboard**: BB\_\*Name\*
  + **Behavior Tree**: BT\_\*Name\*
  + **Behavior Tree Task**: BTT\_\*Name\*
  + **Function Library**: BPFL\_\*Name\*
  + **Macro Library**: BPML\_\*Name\*
  + **Interfaces: BPI\_**\*Name\*
  + **Struct: F\_**\*Name\*
  + **DataTable: DT\_**\*Name\*
  + **AnimationBlueprint: ABP\_**\*Name\*
* **Misc**:
  + **Enumerators**: E\_\*Name\*
  + **Curve (Float)**: Cf\_\*Name\*
  + **Widgets**: WB\_\*Name\*
  + **Texture**s: T\_\*Name\*
  + **Materials**: M\_\*Name\*
  + **Material Instance:** MI\_\*Name\*
  + **Particle Effects:** FX\_\*Name\*
  + **Input Actor:** IA\_\*Name\*
  + **Input Mapping:** IMC\_\*Name\*

# **Changelog:**

### 18/01/2024

V1.0 of General Documentation.

V1.1: Structured the document better by adding more empty space, did another grammar pass, changed the order of some of the chapters and updated setup to include Widget blueprints for AI and Players.

# **Downloads and Set-up.**

## **Download Unreal Engine**

Download the latest version of Unreal Engine 5 from their [website](https://www.unrealengine.com/en-US/download) by downloading the Epic Games Launcher.

## **Download and migrate the project.**

* Download the File folder and extract it.
* Open the uproject and you can work from here, if you want to migrate the project into your own project.
* Open the content browser
* Right click the FirstPersonMelee folder, then click on migrate
* Go to browse through your file explorer where you have your other project.
* Open the Content folder in that project and select okay.

# **Gym**

In “Content/FirstPersonMelee/Maps/Gym” **L\_FPMGym** you can find an example of an Unreal Engine Marketplace package Gym map. This provides similar information to what is in this document but is displayed in a manner akin to how it would be showcased in the Unreal Engine Marketplace. This setup offers a quick refresher on the functions of various Actors and allows you to easily access related assets.

# **DemoLevel**

The demolevel is a level with an AI character using the combat system you can fight to get a feel for the system, you can also spawn in more enemies if you want to make the test more difficult.

# **Video Content:**

If you would rather watch a video explaining how to set up everything in the project you can do so by following this link: <https://youtu.be/XuaccVpOMoY>

You can navigate different parts of the video using the timestamps in the video description.

# **Creating Content**

The following color coding is applied to make scanning easier in the document:  
Panel names  
*Folder Locations*  
Variable names  
Your input should go here [replace the text]  
**Object name**

## How to set up a level

* Go to the right side of the viewport and look for the world settings panel. If you can’t find it click on settings on the right and then click on world settings.
* In the world settings inside your level fill in GM\_FirstPersonGameMode in the GameMode Override tab.
* If you want to use your own game mode but my first person player character, open your game mode and set the player character to BP\_FirstPersonCharacter.

## A screenshot of a computer Description automatically generatedHow to set up an actor.

* The simplest way to set up the system is to use the premade characters you can find in “*Content/FirstPersonMelee/Blueprints/Charact*ers”, or create children from these characters by right clicking and clicking create child blueprint class. And adding your own logic inside these blueprints.
* A screenshot of a computer

  Description automatically generatedIf you want to create/use your own character you can create the actor you want to have use the melee system using convention: **BP\_[YourCharacterNameHere].**

CombatComponent

* Then inside the components panel you will want to add the **BP\_CombatComponent** to the actor by clicking the add button and searching for CombatComponent.
* Once you have added the CombatComponent you can set up a list of variables used for combat, to do this, click on the BP\_CombatComponent inside the components panel, and then look at the right of the screen for the details panel, here you will see a few variables under the Set Up tab.

A screenshot of a computer

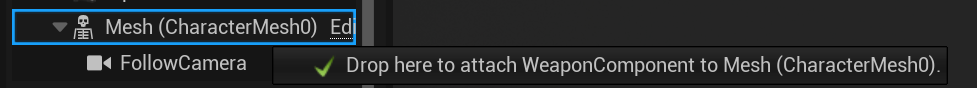
Description automatically generated

* The Health variable: this is your characters health, if it reaches 0 the character will die.
* The CleaveDamage variable: this is how much “damage” is done to an attacks CleaveValue when it hits this character. When the CleaveValue hits 0 the attack can no longer deal damage to other actors.
* The StaggerValue variable: could be seen as a secondary healthbar, when it reaches 0, the character will be immobilized for a short amount of time depending on the attack it was hit by.
* The CleaveStaggerDamage variable: is the same as the CleaveDamage variable but for the attacks StaggerCleave. When this hits 0 the attack can no longer stagger other actors.
* Click on your characters skeletal mesh in the components panel, then in the details panel look for Animation Blueprint, if you want to use my Animation blueprint search for **ABP\_FirstPersonMelee\_C,** also make sure the **animation mode** is set to use animation blueprint

A screenshot of a computer

Description automatically generated

AC\_WeaponComponent

* Then add the **AC\_WeaponComponent** by doing the same thing as the previous step but search for Weapon Component, then drag the weapon component onto your characters mesh. 
* Then in the details panel look for Parent Socket, click on the folder icon to select a Socket, (make sure the location and rotation of the Weapon Component are set to 0 after this.)

A screenshot of a computer program

Description automatically generated

Creating your own socket

you could use one of the hand sockets like MiddleMetaCarpal\_R, or Create your own weapon socket to make sure whatever animations you want to use work properly.

* To do this, go to your characters skeletal mesh and open it.
* Look through the skeletal tree and right click the bone you want to use, like hand\_r and click Add Socket.
* If you want you can also add a preview asset here in order to place the socket properly.
* You can do this by right clicking on the socket and clicking on add preview asset A screenshot of a computer

  Description automatically generated
* To move and rotate the socket to the correct position use the buttons in the top right corner of the viewport or use W to move, E to rotate and R to scale.
* A screenshot of a computer

  Description automatically generated

Input Rotation

* The final component to add to the character. is an arrow component in order to dash. You do this by clicking the add button in the components tab of the character again and searching for “**arrow component**” You can name this arrow component whatever you want but in my setup it is called InputRotation.

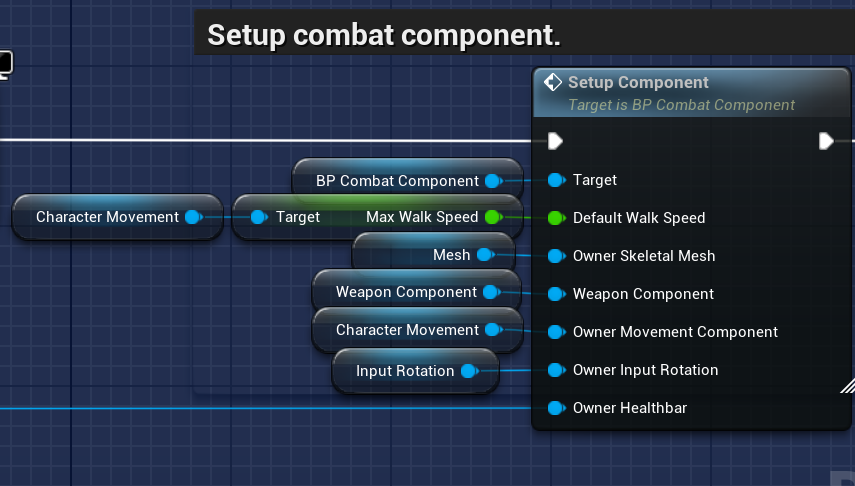
A black and blue background with white text

Description automatically generated

## Setting up the Nodes.

Setup:

To use the Combat component and Weapons logic you will want to create a few nodes. The combat Component needs some variables from the Character it is attached to to work properly,



* Drag off, or create the event begin play by right clicking and then typing Begin Play.
* Then drag off the white arrow and search for Setup Component.
* The target input should automatically get attached to **BP\_CombatComponent**, if it does not make sure to add it to the target input.
* Then for the default walk speed you will want to Either right click and search for Character movement, or drag the Character Movement Component into the blueprint through the Components Panel.
* Then from that variable drag off the Max Walk Speed, and then connect this to the blueprint setup component.
* For the owner skeletal mesh you will want to drag in the Characters Skeletal Mesh.
* For the weapon component you will want to drag in the weapon component.
* For the owner movement component you will once again use the Character movement component.
* For the owner input rotation you will want to drag in the arrow component you just created.
* The owner Healthbar depends on your character, for the player, with my Healthbar you want to drag off begin play and inbetween begin play and Setup component place the Create widget node, the widget class should be WB\_PlayerHUD.
* Off this node drag add to viewport (make sure return value and target are connected.
* Then drag this node into the setup component
* Then use the return value of the create widget node for the owner widget.

A screenshot of a computer

Description automatically generated

* If you want to use the AI healthbar widget you will want to go to the Components panel, and add a widget.
* In the details panel of this widget, set the space value to Screen, and the widget class to WB\_AIHealthBar.
* Also in the details panel, look for collision and set this value to NoCollision

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

* Then, Drag this widget into the blueprint.
* Before connecting this to the owner healthbar, first drag off the widget component variable and look for Get User Widget Object.
* Then off this node you can connect the Owner Healthbar.

A screenshot of a computer program

Description automatically generated

* You can place any other event begin play logic after the setup node.

Inputs

For attacking, if you want to use the aiming logic in the Animation Blueprint for the player, you will want to add some variables and logic to the Player Character. As shown in the picture below, A screenshot of a computer

Description automatically generated

the variables you will want to create are

* Yaw: This gets the left/right axis of the camera.
* Then a Min and Max yaw clamp variable, to limit how far the camera can rotate, I set the Min Yaw Clamp value to -10 and the max to 20.
* The same with the Pitch except this will control the up and down axis.
* I set the Min Pitch Clamp to -40 and the Max to 40.
* To create variables look at the My Blueprint panel, this should be under the components panel. And click on the plus icon of the variables tab, for these variables you will want to use float variables.
* You can change the variable types by clicking on the drop down next to the variable name.

A screenshot of a computer

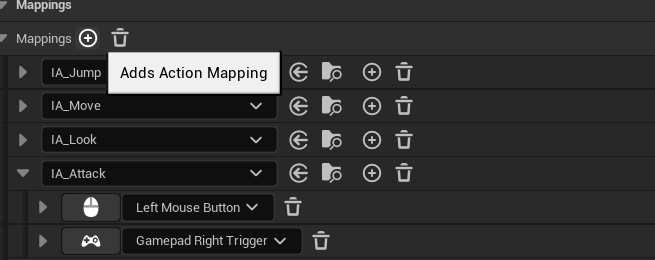
Description automatically generatedA screenshot of a computer

Description automatically generated

Creating an input action.

A screenshot of a computer

Description automatically generatedThen you will want to add your attack input to the blueprint, if you do not yet have an attack input you can either use the Input mapping context provided by changing the mapping context in your character. Or adding a input to your existing one. To do this the easiest way I recommend duplicating the Jump input action, and then renaming this to IA\_[Attack] and then adding this to the IMC by going to “content/input/IMC\_Default.”

A screenshot of a computer

Description automatically generated

Then off this input, drag “Use Weapon” off Started and “Stop Using Weapon” off Completed. A screenshot of a computer

Description automatically generated

For Blocking you will do the exact same, Create a blocking input if you do not have one, then off started drag Use Block, off Completed drag Stop Block. A screenshot of a computer

Description automatically generated

For Dashing you will once again create a dash input, and then drag Use Dash off Triggered.

A screenshot of a computer

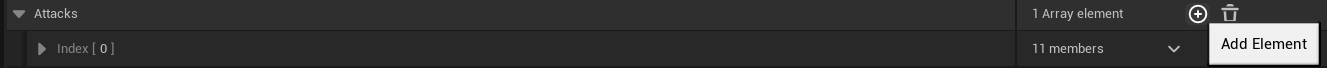
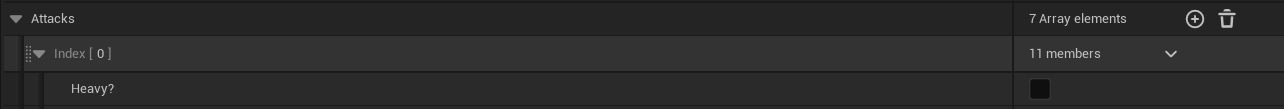
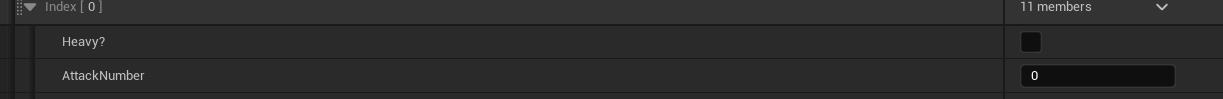
Description automatically generated

The setup for an AI to use the system would be the same, Except instead of input actions you would use blackboard actions or a different system, I will not go over this here as the system is not about AI, just know that it is also possible to let the AI use the system. Simple examples can be found in the Gym Level and demo level.

## Creating a weapon

* To begin making a new weapon, first go to BP\_MeleeWeaponBase in *“Content/FirstPersonMelee/Blueprints/Weapons”* Right Click on the blueprint and create a child actor called **BPC\_[YourWeaponNameHere],** Leave this blueprint for now.
* Next, go to *“Content/FirstPersonMelee/Blueprints/Structs”* and open DT\_WeaponTable.
* If you want to create a weapon from scratch click on the Add button in the top left.A screenshot of a computer

  Description automatically generated
* You can also use an existing weapon as a base and tweak it’s stats by right clicking on the weapons’ row you want and duplicating it.A screenshot of a computer

  Description automatically generated
* When you create a new weapon, click on the row you just created and press F2, or right click and Rename, Rename the row to something simple as this will be needed later. You can fill in the same name in the weapon name slot.
* The next slot: IdleAnimation, is what animation is played when the character is holding the weapon, this would mainly be used to distinguish between 1-handed and 2-handed weapons as an example.
* Similarly to IdleAnimation, the block animation is what **anim montage** to play when the character is blocking, which might differ between different weapon types.
* The variable BlockRadius: Is the radius in which the weapon can effectively block in degrees, so if this is 360, the weapon can block every attack even if attacked from behind, or 30 would only really block in front of the player.
* Whenever the player gets hit while blocking it will drain BlockStamina, this amount will differ per weapon which we will get into later, here you set the maximum amount of block stamina the weapon has, the higher the value, the more hits can be blocked before you are guard broken.
* The variable EffectiveDashes is similar to BlockStamina, this variable is basically how many dashes can be spammed before the charges have to regenerate.
* The Variable DashIntensity is how far and fast the dash of this weapon is.
* Next up is the variable Attacks, this one is slightly different as this is a struct array within the struct, click on the + button to add an attack. 
* You can add as many attacks as you want, both light attacks (click attack) or heavy attacks (hold attack), to decide whether or not an attack is heavy or light you will want to check the Heavy? Variable. 
* The amount of light and heavy attacks. do not need to be the same amount of attacks, so you could have 3 light attacks and 4 heavy attacks for example. You will need to fill in the MaxLightCount and MaxHeavyCount below the attacks variable, these variables will the the highest AttackNumber of both your light and heavy attacks respectively. This value starts at 0. So your first attack would have attack number 0, your second 1, your third 2, etc. 
* Do make sure that you count this for both heavy and light attacks, so both your first heavy and light attack would have AttackNumber 0.
* The Animation variable is what Anim Montage you want to use for each attack, If you want to use your own Animations, you will need to do some work in order to make the animations work properly, please take a look at the Setting up Animations chapter in this doc.
* The Speed variable is a modifier that makes the attack animation play faster or slower, which in turn influences the attack speed.
* The BaseDamage value is the amount of damage the attack deals.
* The StaminaDamage value is the amount of block stamina this attack drains when the attack is blocked.
* The CleaveLimit variable could be seen as a healthbar for the attack, whenever you hit an enemy they deal CleaveDamage to the weapon, if the CleaveLimit of the attack reaches 0 it can no longer deal damage, so the higher the cleave limits, the more enemies can be hit at the same time.
* The Knockback value is how far the enemy you hit with this attack will be knocked back. The higher the number, the further and higher they are knocked back.
* The StaggerDamage value is the amount of damage the attack deals to the enemies StaggerValue, this is like a healthbar, but when this value reaches 0 enemy hit is immobilized for the StaggerDuration of the attack.
* The StaggerCleaveLimit functions exactly the same as the CleaveLimit variable, but for the stagger value.
* The final value in the Data Table is AttackRangeModifier. This allows you to increase or decrease the range of the weapons hitbox.

Speaking of the weapons hitbox. To set up your weapons mesh you need to add 2 sockets, the Start and End sockets.

* To do this, go to the mesh you want to use for your weapon and open it.
* On the right side open the Socket Manager Panel.A screenshot of a computer

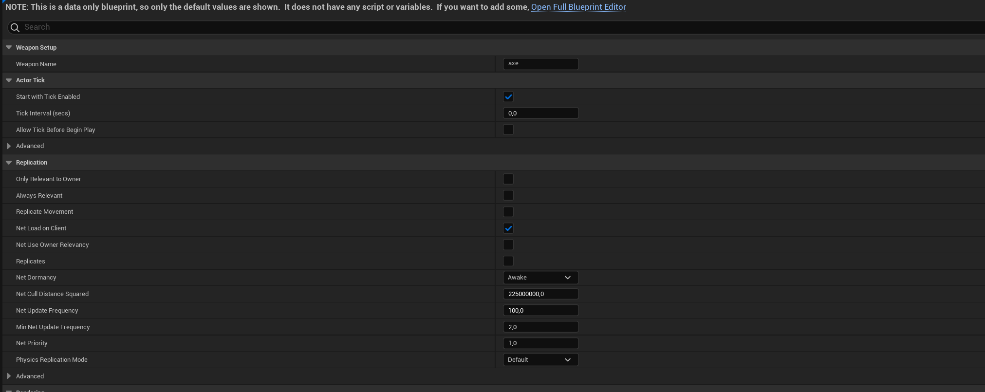
  Description automatically generated
* Then click the + button to create the 2 sockets named “Start” and “End”.
* A blue object in front of a building

  Description automatically generatedMove these sockets using the arrows to the positions of where you want the damage hitbox to start and end.

Now to finish creating your weapon:

* Open the BPC\_[YourName] Blueprint again.

You should get a screen like this:



In here type in the row name of your weapon in the Weapon Name slot.

* If you didn’t get this screen but instead the full blueprint editor you can find this variable in the Details Panel on the right.A screenshot of a computer

  Description automatically generated
* If you were in the other vier before, at the top, click open full blueprint editor, then at the top left, click on the viewport tab. A screenshot of a computer

  Description automatically generated
* A screenshot of a computer

  Description automatically generatedGo to the left and click on WeaponMesh in the Components Panel, then in the details panel Replace the static mesh with whatever mesh you want to use for this weapon, and also set the correct material if you have to. (the correct material can be found in the static mesh tab where we set up the sockets, you can also scale or move the weapon mesh in the details panel if you have to.

Then if you have done this, the weapon should be set up, now you can drag it into a level and pick it up to test it, you can do this by holding left click and dragging the blueprint from the content folder into the level viewport.

## Setting up Animations.

* A screenshot of a computer

  Description automatically generatedTo create an Anim Montage for attack animations, right click on the animation sequence, go to create, anim montage. Once inside the montage you will want to set up the montages slot, to do this go to DefaultGroup.DefaultSlot next to the animation track and click on the down arrow, go to slot name and select upper body or full body depending on what type of montage you want to use, for anything that would require you to still be able to move I recommend the upper body slot. A screenshot of a computer

  Description automatically generated
* If the animation “breaks”, that is normal, just save, close and reopen the montage.
* If this is an attack animation, the next step is to right click on the 1 track, if you do not have the 1 track, go to the left of the track, click on the notifies down arrow and select add notify track. A screenshot of a computer

  Description automatically generated
* When you right clicked on the 1 track select add notify state and look for AnimNS\_HitDetection.A screenshot of a computer

  Description automatically generated
* Once you have added this notify state you want to drag the diamonds to where you want the weapon to start damaging, to where you want it to stop damaging. A screenshot of a computer

  Description automatically generated
* Then also on the 1 Notifies track, add notify and add, CanAttackAgain. A computer screen shot of a robot

  Description automatically generated
* If you do not have this notify, you could copy it from a Anim Montage provided in “*FirstPersonMelee/Animations/Montages”* or create it yourself by clicking New Notify. **MAKE SURE IT IS CALLED “CanAttackAgain”.** I recommend placing the notify at the end of the HitDetection notify, however, if you want to have more deliberate combat like in Dark Souls you can put this notify at the very end of the animation.

## Misc

This project is a base, not a finished combat system. I encourage you to look at the code inside BP\_CombatComponent and BP\_WeaponBase and tweak them to fit your own project.