UNREAL ENGINE 4

MVP SERIES

FPS

#1 Introduction

GOAL FOR THIS LESSON

We will be going over the project and having a look at what we will be making. This will be a basic prototype of a FPS game and we will be starting from scratch using a blank project. We will only be using items that come with the blank project and adding nothing so this will be a good starting point for games in the future.

WHAT IS GOING ON

- Overview of the Completed Project
- Discussion of what we will be learning

DID YOU LEARN?

- Did you notice how almost all of the systems we will create are generic enough to expand upon later? This will allow us to create a good basic template to build upon.

TRY THIS (optional)

- Create a folder structure that we will be using for the rest of this series. We will have Images, Blueprints, a Character, and a few other things.

#2 Basic Setup

GOAL FOR THIS LESSON

We will be creating the basic folder structure for our game so we can try and keep it organized. After that we will start on getting the basic classes created that we will be using and then tackling the movement of our character.

WHAT IS GOING ON

- Disabling Play in Editor sounds
- Creation of the basic folder layout
- Creating our new Game Mode
- Creating our Character Blueprint
- Creating our Player Controller Blueprint
- Saving our work so far
- Setting up our input so we can move with our WASD Keys
- Creating the Events that will let us move
- Setting up the input so we can look with our Mouse
- Creating the Events that will let us look
- Fixing the inverted Mouse Look
- Fixing the problem with moving while looking down.

DID YOU LEARN?

- Pawn and Character are similar Actor Classes but only one comes with almost everything you need to let the player move it around your game.
- Did you learn the difference between an Action and Axis mappings in Input?

TRY THIS (optional)

- Add a new input called Jump that is linked to the Spacebar.
- Add a bool variable and an If node in the Controller to allow us to easily use Inverted Mouse Look.

#3 Setting Up The Crosshair

GOAL FOR THIS LESSON

We will be setting up the Crosshair using UMG (Unreal Motion Graphics) so the player knows where they are aiming when we fire.

WHAT IS GOING ON

- Creation of the new folder
- Creating our new Widget Blueprint
- Setting up an Image for the Crosshair
- Accessing Engine Content
- Finding a good Crosshair
- Creating our new Function
- Hooking up the new Function

DID YOU LEARN?

- Even the basic version of the Editor comes with hundreds of items already created for our use.
- Why did we create the Crosshair in the Player instead of the Controller?

TRY THIS (optional)

- Try out other options for a Crosshair using the default Engine content.

#4 Setup Our Line Trace

GOAL FOR THIS LESSON

We will be setting up our Line Trace so we can simulate firing a simple projectile weapon. We will use debugging to verify it is working but not actually hooking anything up to the event for hitting anything yet.

WHAT IS GOING ON

- Setup our Fire Key in the input settings
- Hook up the Event for firing in the player controller
- Create a Player BPI for handing the Fire Event
- Set the Player to use the new BPI
- Code the Line Trace in the Player
- Adding our Camera to the Player
- Fixing the Character Rotation problem
- Finishing the Line Trace

DID YOU LEARN?

- Why are we setting up the input event in the Controller but having the Player fire off the event itself?
- Why did we need to have the Character inherit the rotation from the controller?

TRY THIS (optional)

- Change the Line Trace Channel from Visibility to Camera and see what happens.

#5 Setup Our Hit Event

GOAL FOR THIS LESSON

We are going to finish our Line Trace event by shooting at Targets. First we will finish up the Line Trace to have it check for hitting something then we will create a Blueprint for a Target and have it destroy itself when hit.

WHAT IS GOING ON

- Examining our Hit Result structure
- Checking to see if we hit anything
- Creating a Generic Blueprint Target
- Setting up the Target Blueprint Interface
- Creating the Event for handling being hit
- Testing and destroying the Target

DID YOU LEARN?

- Why did we test the same basic thing multiple times as we were adding in new code?
- Why do we want the target to handle the death event rather than the player who shot it?

TRY THIS (optional)

- Add in a Delay to prevent our player from firing more than 1 time a second.
- Create another Target Blueprint but this time use a sphere for the shape.

#6 Setup Our Score System

GOAL FOR THIS LESSON

We are going to setup a scoring system so that when we hit the targets we get some points and display it on the HUD.

WHAT IS GOING ON

- Updating our original HUD Blueprint
- Creating a new Widget for the new HUD
- Setting up the new HUD Widgets
- Creating our Function to Update the Score
- Displaying our HUD Widget
- Adding our Update Score Event to the Controller BPI
- Updating our Targets HandleWasHit Event
- Fixing any Bugs

DID YOU LEARN?

- Why are we not binding the HUD Text Block to the Players Score directly and instead using a function to push the change to the HUD?

TRY THIS (optional)

- Adjust the score on the HUD to show 6 digits at all time by using the Padding Node in the Update Score event on the HUD. For example it should show 000500 if they have 500 for the score.

#7 Give Values To Our Target

GOAL FOR THIS LESSON

We will be givng our Targets some health so we can shoot them more than one time. In addition we will give a bonus to the score when we finally destroy them by depleting all of their health.

WHAT IS GOING ON

- Setting an exposed variable for Health and Score on the Target
- Updating the Target getting hit to take damage
- Organizing our Inputs
- Finishing up taking damage
- Give the player a Bonus when they destroy the target
- Expose the Player Damage Variable

DID YOU LEARN?

- Why are we setting the variables for Health and Damage to be exposed rather than setting them in the Blueprint itself?
- Why are we using a <= (Less than or Equal) node instead of an == (Equals) node?

TRY THIS (optional)

- Setup a Right Mouse Click that will do more damage than the Left Mouse Click so you can destroy the targets quicker but dont let it fire more than once every 3 seconds.

#8 Custom Materials

GOAL FOR THIS LESSON

We will be creating a custom material for the Targets to show them Green when they are full health and changing in color to Red when they are out of health. We will also be adjusting the Crosshair material so we can see thru the parts that are black instead of having them be solid and blocking our view.

WHAT IS GOING ON

- Setting up our Folders and Material
- Assigning the Material to our Target
- Creating a Basic Color Material that we can adjust
- Setting up the new Dynamic Material on our Target Blueprint
- Creating a Function to Set our Color at runtime
- Testing our new Material and fixing Bugs
- Fixing the Crosshair

DID YOU LEARN?

- Why did we have to create a Dynamic Material Instance for the Target when we already had a Material assigned?
- Whydid we have to create a new Material for the Crosshair to allow it to be partially Transparent?

TRY THIS (optional)

- Adjust the Lerp on the Set Color function in the Target so that it starts at green and goes to Red when there is 1 health left instead of it being partially orange at 1 health left.

#9 Adding A Particle Effect

GOAL FOR THIS LESSON

We will be creating a small particle effect in the Particle Editor to simulate us hitting something when we fire our weapon.

WHAT IS GOING ON

- Setting up the new Particle we will be using
- Creating the Material we will be using in the Particle
- Adjusting the Particle settings to make pixel smoke
- Adjusting our Fire Action to spawn the particle

DID YOU LEARN?

- Particles are super complicated and evil and have been known to cause black holes when created improperly.
- Why are we spawning the particle in the fire event rather than having the Target spawn it when hit?

TRY THIS (optional)

- Play with the particle to add in some color and have it go from a red to yellow to black rather than the black to white we are using.

#10 Game Over Screen

GOAL FOR THIS LESSON

We will be creating a new UMG Widget that will handle displaying a simple game over screen. It will allow us to Restart or Quit the current game as well as show us how many points we scored in the last game.

WHAT IS GOING ON

- Creating our new Widget for Game Over
- Laying out the Game Over Screen
- Adjusting our Game Mode to track Targets
- Adjusting our Game Mode to handle Game Over
- Adjusting our Targets to talk to the Game Mode
- Playtesting and Bug Fixes
- Making the Game Over screen talk to the Controller
- Adding in Quit and Restart Buttons
- Playtesting and Bug Fixes
- Handling Starting a new Game in the Game Mode

DID YOU LEARN?

- What is the difference between the 3 Input Modes for the player controller?

TRY THIS (optional)

- Make the Game Overscreen auto restart if the player does not click anything for 10 seconds.

#11 Conclusion

WRAPPING IT ALL UP

So here we are at the end of the series and hopefully you have been able to create a basic FPS and learned a few new things. Now that we have our basic system we will be able to expand on this in the future with other modules and flesh it out a bit with some more features.

THINGS TO TRY

- Create a Start Screen that the game loads up first rather than right into the game
- Introduce Enemies that fight back
- More than 1 Weapon
- Sound Effects
- Models

Special Thanks

A special Thank You to the following people

- **Epic Games** - Making this awesome engine, making Blueprints a first class citizen of it, and allowing us all to play with it
- **Kitatus** - Introducing me to the awesome world of UE4 Blueprints and Tutorial Creation
- **s-ed** - Awesome person who helped with ideas to flesh out the presentation of this series
- **The Family** - Giving me 2 hours a day to focus and get this done...
- **Unreal Slackers** - Awesome group of people to bounce stuff off of with only minimal name calling