Unreal Engine Course Designed By Ethan Minnich

Module 1 Goals:

- **Familiarize with Unreal Engine 5's basic tools and interface.**
- **Initiate your first project and design a custom level from scratch.**
- **Construct walls and floors by utilizing meshes.**
- **Incorporate actors and meshes from content packs to craft unique environments.**
- **Apply Transform Tools to create more intricate objects and shapes.**
- **Modify materials and textures on meshes to customize the appearance.**
- **Arrange various objects and meshes to develop a platforming challenge.**
- **Ensure surfaces and meshes have proper collision settings.**

Create a Project

- **Launch Epic Games and sign in. If you don't have an account, create one.**
- **Create a Project: Ensure it's set to Third Person with Blueprint and Starter Content.**

Controls:

- **Left-click: Select objects.**
- **Right-click and drag: Look around the scene.**
- **WASD keys with right-click held down: Move around the scene.**
- **Scroll wheel: Zoom in and out.**
- **Alt + P: Start playtesting.**
- **Esc: Stop playtesting.**
- **Ctrl + Shift + S: Save your project.**

Change Frame Rate:

- **Go to Settings.**
- **Click on Engine Scalability Settings and change it from Epic to Medium.**
- **Navigate to Edit > Project Settings in the top-left corner.**
- **Under Engine > Rendering, search for "Lumen" and set Dynamic Global Illumination to None.**
- **Under Reflections, set the Reflection Method to None.**
- **Search for "Shadow" under Engine Rendering > Shadows and set Shadow Map Method to Shadow Map.**
- **Search for "Anti" and under Default, set Anti-Aliasing Method to Fast Anti-Aliasing.**

Add Content Pack:

- **Save your project.**
- **Close UE5 and open the Epic Games Launcher.**
- **Select Marketplace and choose the Free drop down.**
- **Search for building or any asset pack you want!**
- **Click the Free button to download.**
- **Then Click Add To Project and Select Your Game**

Create a Level

- **Open Unreal Engine and click Play.**
- **Move around the scene using WASD.**
- **Go to File > New Level > Basic and create a new level.**
- **Save the level via File > Save Current Level and name it CastleLevel.**
- **Open the level with File > Open Level.**

Set Default Level

- **Go to Edit > Project Settings.**
- **Under Project, select Maps and Modes.**
- **Set the Editor Startup Map to your desired level.**

Navigate a Level

- **Rotate: Hold Right Click and drag the mouse.**
- **Move: Hold Right Click and use the WASD keys.**
- **Pan Up/Down: Hold Right Click and press Q or E.**

Reset the Viewport:

- **In the Outliner, click on an object and press F.**

Save Level:

- **Ctrl + S: Save the current level.**
- **Ctrl + Shift + S: Save all levels.**

Create a Wall:

- **Go to the Content Browser.**
- **Click on the Add an Asset Filter and select Static Mesh.**
- **Ensure the Content folder is selected, and type Wall in the search bar.**
- **Click on SM_Plains_Wall_Straight_01 and drag it into the viewport.**

Remove Static Meshes

- **Select the object and hit the Delete key.**

Move an Object:

- **X — horizontal (width)**
- **Y — vertical (height)**
- **Z — perpendicular (depth)**
- **Select the wall and press W for the Translate Tool; drag the widget to move the wall.**
- **Press R to scale and drag the widget to resize the wall.**
- **Press E to rotate and drag the circle to rotate the wall 90 degrees.**

Transform Tools:

- **In the top-right, next to Translate, Scale, Rotate, experiment with different snapping settings.**

Expand the Wall:

- **Navigate to Content > InfinityBladeGrassLands > Environments > Plains.**
- **Create different styles of walls using various static meshes.**
- **Hold Shift and select multiple static meshes in the Outliner, then press Ctrl + G to group them (or right-click and group). This allows you to move the entire group together.**

Duplicate Walls:

- **Select a wall, press Alt, and move the widget to duplicate it.**
- **Use static meshes to build a castle.**

Materials:

- **Clear any searches in your Content Browser and disable any filters.**
- **Navigate to the Content folder.**
- **Click the Filters button and select Material.**
- **Drag and drop materials to change them from checker patterns. Use the Detail Panel to further customize the material.**

Player Start:

- **Open your level in UE5.**
- **Locate and delete any existing Player Starts in your level.**
- **In the Place Actors panel, select the Basic tab.**
- **Click and drag Player Start into your viewport.**
- **You can rotate and place it anywhere in your level.**

Level Planning:

- **Create an image folder in your student folder for quick access to reference images.**
- **Download reference images from online sources and save them to your folder.**
- **Draw your level map from a top-down perspective, using these image references as a guide.**

Build a Platforming Challenge:

- **Build a platforming game where the player must jump on obstacles to reach the top.**
- **Use static meshes creatively to design the challenge.**
- **Adjust collisions as needed.**
    - **Click on your static mesh in the Content Browser to open the Static Mesh Editor.**
    - **Select Collision > Remove Collision > Set Collision to Auto Convex Collision.**
    - **In the Convex Decomposition panel at the bottom-right, click Apply.**
    - **Hit Save.**

Module 2

Goals

- **Design a landscape with grass and trees.**
- **Create custom materials and apply them to the landscape.**
- **Display a name (string) on the screen when a key is pressed using Blueprints.**
- **Create and store a string in a variable.**
- **Use integers to track a player's gold and how much they collect.**
- **Add comment boxes, reroute pins, and organize nodes using functions.**

Create Landscapes

- **Open your CastleProject and delete the floor beneath your castle.**
- **Switch to Selection Mode > Landscape.**
- **In the Landscape panel, click the Section Size drop-down and select 15x15 Quads (or choose 31x31 Quads for a larger map).**
- **Click Create at the bottom to generate the new landscape.**

Sculpt Tool

- **Click and drag across your landscape to raise it.**
- **Hold Shift while dragging to create valleys.**
- **In the Landscape panel, adjust the Sculpt Tool settings as needed.**

- **Use the Sculpt Tool to form a mountain range around the castle and a valley within it.**
- **Continue sculpting to create a unique landscape for your level.**

Landscape Materials

- **In the Content Browser, right-click in the Content folder and select New Folder.**
- **Name the folder "CustomMaterials".**
- **Open the folder, right-click inside, and select Material. Name it "MAT_Landscape".**
- **Double-click the material to open it in the Material Editor.**
- **Click the Restore Down button in the top-right corner to prevent the window from taking up the entire screen.**

Textures

- **In the Content Browser, navigate to the Content folder and apply the Texture filter.**
- **In the search box, type "_D" to filter out textures that aren't currently relevant.**
- **Find two textures you like and drag them into the Material Editor window.**

Material Graph

- **Maximize the Material Editor window.**
- **Drag the two Texture Sample nodes to the left to avoid overlap with other nodes.**

Blend Layers

- **Right-click in the Material Graph, search for "LandscapeLayerBlend", and add the node.**
- **Place the node between the Texture Sample nodes and the MAT_Landscape node.**
- **In the Details panel of the LandscapeLayerBlend node, click the + next to Array Elements for each texture you added.**
- **Rename the Layer Names to match your textures (e.g., "Grass" and "Gravel").**
- **Set the Preview Weight of the texture you'll use most to "1", leaving the others at "0".**
- **Connect the RGB output pins of the texture nodes to the appropriate layer input pins.**
- **Connect the output pin of the LayerBlend node to the Base Color input pin on the main MAT_Landscape node.**

Roughness

- **Press 1 and click in the Material Graph to create a Constant node.**
- **In the Details panel, set the Value to "1".**
- **Connect the Constant node's output to the MAT_Landscape node's Roughness input pin.**

Landscape Layer Coords

- **Right-click the empty space to the left of your texture nodes and search for "LandscapeLayerCoords" to add the node.**
- **Select the Landscape Coords node and, in the Details panel, change the Mapping Scale value to "7" to scale up your textures.**
- **Connect the output pin of the LandscapeLayerCoords node to the UVs pins on the texture nodes.**

Save the Material

- **Click the Save button in the top-left corner to save your Material Graph.**
- **Minimize the Material Graph to return to your UE5 project.**

Paint Landscapes

- **Switch back to Selection Mode.**
- **In Selection Mode, select your landscape.**
- **Locate the "MAT_Landscape" material in the Custom Materials folder in the Content Browser.**
- **In the Details panel, expand the Landscape section.**
- **Drag the "MAT_Landscape" material from the Content Browser into the Landscape Material value box in the Details panel.**

Weight-Blended Layers

- **In Landscape Mode, go to the Paint tab in the Landscape panel.**
- **Under Target Layers, locate your materials.**
- **Click the + next to the layer and select Weight-Blended Layer.**
- **In the Create New Landscape Layer Info Object window, click Save.**
- **Repeat these steps for all your layers.**

Paint the Land

- **In the Landscape panel, under Layers, select the layer you want to paint.**
- **Click and drag your brush across the areas you want to paint, similar to sculpting.**

Transition Between Textures

- **Adjust your brush size and tool strength to create smooth transitions between textures.**
- **Experiment with different settings to refine your landscape.**

Foliage Tool

- **Open your level in UE5 and find an empty area on the landscape.**

- From the Modes selection, choose Foliage to open the Foliage panel.
- In the Content Browser, search for "tree" Static Meshes.
- Drag and drop a tree into the Drop Foliage Here section of the Foliage panel.
- Repeat for 1-2 more Static Meshes.

Paint the Trees

- Select the Static Mesh you want to edit.
- Set the Density to a value between 1 and 10.
- Drag across the level to paint a small test strip and check your settings.
- Adjust as needed and continue painting your landscape with trees.

Intro to Blueprints

- In the Content Browser, navigate to the All > Content > ThirdPerson > Blueprints folder.
- Double-click the BP_ThirdPersonCharacter Blueprint to open it in a new window.

Open Level Blueprints

- Minimize the Third Person Character Blueprint window.
- In the toolbar, click the Blueprints button and select Open Level Blueprint.

String Variable Blueprints

- Click the Blueprints button and choose Open Level Blueprint.
- Right-click in the Event Graph and search for "keyboard n".
- Select N under Keyboard Events to add the node.

Print a String

- Drag from the Pressed output pin and release the mouse button.
- Search for "print string" and select Print String (under Development) to add the node.
- In the In String box, type the text you want to display, like "Hello player".
- Click Compile in the top-left corner and save your project.
- Minimize the Blueprints window and click Play to test it.
- Press the N key to see the text in the top-left corner.

Create a Variable

- Open the Level Blueprint window.
- In the My Blueprints panel, click the + next to Variables and name it "PlayerName".
- In the Details panel, set the Variable Type to String.
- Click Compile in the top-left corner to save your work.

- **Under Default Value in the Details panel, type the name you want to display instead of "player," like "Warbler".**

Display the Variable

- **Drag the "PlayerName" variable into the Event Graph and select Get PlayerName.**
- **Connect the Player Name node's output pin to the Print String node's input pin.**
- **Compile and save your Blueprint.**
- **Minimize the Blueprints window, click Play, and press N to test it.**

Append the Variable

- **Alt + click the wire connecting the Player Name and Print String nodes to remove it.**
- **Drag from the Print String node's In String input pin and search for "append string".**
- **Add the Append node (under String) and type "Hello " in the A input box.**
- **Connect the Player Name node's output pin to the B input pin of the Append node.**
- **Compile and save your Blueprint. Then, minimize the Blueprints window, click Play, and press N to test it.**

Integers & Branch Blueprints

Integer Strings

- **Open the Level Blueprint in your UE5 project.**
- **Right-click and search for "keyboard g", then select G Keyboard Event.**
- **Drag from the Pressed output pin and search for "print string" to add a Print String node.**

Create Integer Variables

- **In the My Blueprint panel, click the + next to Variables and name it "TotalGold".**
- **Set the Variable Type to Integer.**
- **Repeat the process to create a new integer variable named "GainGold" to represent the gold received.**
- **Compile and Save.**
- **In the Details panel, set the Default Value for the "GainGold" variable to "1".**

Set the Variable

- **Drag the "TotalGold" variable into the Event Graph and select Get TotalGold.**
- **Repeat for "GainGold".**
- **Drag from the "TotalGold" output pin, search for "plus", and click Add.**
- **Connect the "GainGold" output pin to the bottom Add input pin.**
- **Drag from the Add output pin, search for "set total", and select Set TotalGold.**

- **Connect the G Pressed output to the Set input.**
- **Drag from the Set output to the Print String input.**

Print a Variable

- **Drag from the In String input, search for "append", and select Append.**
- **In the A input, type "You collected ".**
- **Drag the "GainGold" variable into the Event Graph and select Get GainGold.**
- **Connect the Gain Gold output to the Append B input.**
-

Branch Nodes

- **Drag from the Print String output, search for "branch", and select Branch.**
- **Drag from the Condition input, search for "greater equal", and click Greater Equal.**
- **Drag from the Add Pins**
- **Click Add pin + to add a C input.**
- **Type " gold. You now have " in the C input.**
- **Add D and E inputs by clicking Add pin + twice.**
- **Drag the "TotalGold" variable into the Event Graph and select Get TotalGold.**
- **Connect the Total Gold output to the Append D input.**
- **Type " gold. " in the E input.**
- **Compile, Save, minimize the Blueprints window, click Play, and press G to test your game.A input, search for "get total", and select Get TotalGold.**
- **In the B input, type a number like "10".**

True or False Outputs

- **Drag from the Branch node's True output, search for "print string", and select Print String.**
- **Type a message like "You have a lot of gold!" in the In String field.**
- **Repeat for the Branch node's False output, with a message like "You don't have enough gold."**
- **Compile, Save, minimize the Blueprints window, click Play, and press G to test your game.**

Organize Nodes

- **In your UE5 project, open the Level Blueprint you've been working on.**
- **Click and drag to highlight all the nodes you've added in previous lessons.**
- **With all nodes selected, press C to add a comment box.**
- **Name the group "Blueprints Logic Practice" and press Enter.**
- **Highlight the nodes from the previous lesson and create a new comment box inside the larger one, naming it "Gold Collection".**
- **Optionally, change the Comment Color in the Details panel.**

Organize Wires

- Select a wire you want to reroute.
- Double-click the wire to add a reroute point, and drag the point to reposition the wire.

Create a Function

- Select the Set, Add, and variable nodes.
- Right-click one of the selected nodes and choose Collapse to Function.
- Name the function "AddGold" and press Enter.
- Double-click the Add Gold node to open it in a new tab.
- Rearrange your wires, then Compile and Save.

Module 3 Goals

- Create Blueprint Classes for a key and door.
- Utilize Branch nodes to check for variable conditions.
- Design a Blueprint that launches the player into the air.
- Enable double-jumping by increasing Max Jump Count.
- Create a collectible coin that is added to the player's inventory.
- Build a UI Widget to display the coin count.
- Develop a sprinting system using inputs and functions.

Locked Door and Key Setup

- Open your project in UE5 or create a new one.
- In the Content Browser, navigate to All > Content > ThirdPerson > Blueprints folder.
- Right-click and choose Blueprint Class.
- In the Pick Parent Class window, select Actor.
- Name it BP_Key.
- Repeat steps 3-5, but name the second one BP_Door.

Create the Key

- Double-click BP_Key to open it.
- In the Components panel, select Add > Sphere and name it Key.
- In the Details panel, under Materials, select a colored material like M_Metal_Gold for the key.

Add the Trigger Box

- **In the Components panel, select the DefaultSceneRoot.**
- **Select Add > Box Collision and name it TriggerBox.**
- **Use the Scale tool (R) to resize the box so that it's larger than the sphere.**
- **Minimize the Blueprint and return to the UE5 level.**

Create the Door

- **Double-click BP_Door to open it.**
- **Add a Static Mesh and name it Door.**
- **In the Details panel, under Static Mesh, use the drop-down and type "door" to find and select SM_Door.**
- **In the Components panel, select DefaultSceneRoot and add a Box Collision named TriggerBox.**
- **Use the Scale tool (R) to resize the box so it's larger than the door, and use the Move tool (W) to position it around the door.**
- **In the Components panel, select DefaultSceneRoot and add another Box Collision named CollisionBox.**
- **Scale and move this box to match the door's size.**
- **In the Details panel, under Collision, click the Collision Presets drop-down and select BlockAll.**

Locked Door Puzzle with Blueprints

Key Blueprint

- **In the BP_Key, select the Event Graph tab.**
- **In the Components panel, right-click TriggerBox and select Add Event > OnComponentBeginOverlap.**
- **Drag from the Exec output pin and add the Cast To BP_ThirdPersonCharacter node.**
- **Connect the Other Actor output pin to the Object input pin.**

Create the Key Variable

- **In the Content Browser, open BP_ThirdPersonCharacter Blueprint and select the Event Graph tab.**
- **In the My Blueprint panel, click the + next to Variables to create a new variable.**
- **Name it Key and set its type to Boolean.**
- **Click Compile and Save.**

Set the Variable Condition

- **Go back to the BP_Key tab.**
- **Drag from the As BP Third Person Character output pin of the Cast To BP_ThirdPersonCharacter node and add the Set Key variable node.**

- **Connect the Exec output pin of the Cast To BP_ThirdPersonCharacter node to the Exec input pin of the Set node.**
- **Check the box next to Key to set the variable to true.**

Destroy the Object

- **Drag from the Exec output pin of the Set node and add the Destroy Actor node.**
- **Click Compile and Save, then minimize your Blueprints.**
- **Drag the BP_Key into your level and resize it as needed.**
- **Click Play and test that the key is destroyed when the player character collides with it.**

Door Blueprint

- **In the BP_Door Blueprint, select the Event Graph tab.**
- **In the Components panel, right-click TriggerBox and select Add Event > OnComponentBeginOverlap.**
- **Drag from the Exec output pin and add the Cast To BP_ThirdPersonCharacter node.**
- **Connect the Other Actor output pin to the Object input pin.**

Set Branch Conditions

- **Drag from the Exec output pin of the Cast To BP_ThirdPersonCharacter node and add the Branch node.**
- **Drag from the As BP Third Person Character output pin of the Cast To BP_ThirdPersonCharacter node and add the Get Key variable node.**
- **Connect the Boolean output pin from the Get Key node to the Condition input pin of the Branch node.**
- **Drag from the True output pin of the Branch node and add the Destroy Actor node.**
- **Drag from the False output pin of the Branch node and add the Print String node.**
- **For the In String value, type "Door is locked."**
- **Test the functionality by compiling, saving, and playing the game.**

Build a Jump Pad with Blueprints

Create a Blueprint Class

- **Open your UE5 project.**
- **In the Content Browser, navigate to Content > ThirdPerson > Blueprints folder.**
- **Right-click and select Blueprint Class to create a new one.**
- **Select Actor.**
- **Name it BP_JumpPad.**

Add a Component

- **Double-click BP_JumpPad to open the Blueprint Editor.**
- **Select the Viewport tab.**
- **In the Components panel, click Add Component.**
- **Type "Cylinder" and select it to add the component.**
- **Use the Scale tool to adjust the cylinder's size for your jump pad.**

Script with Blueprints

- **Click the Event Graph tab.**
- **In the Components panel, select your Cylinder Component.**
- **In the Details panel, scroll down to the Events section.**
- **Click the + button next to On Component Hit to add the event.**

Cast to Third Person Character

- **Drag from the white execution pin of the On Component Hit (Cylinder) node and release the mouse button.**
- **Search for "Cast to third person character" and select Cast To BP_ThirdPersonCharacter to create that node.**

Launch the Player

- **Drag the Other Actor output pin to the Object Input pin.**
- **Drag from the execution pin of the Cast To BP_ThirdPersonCharacter node and release the mouse button.**
- **Search for "launch character" and select Launch Character.**
- **Connect As BP Third Person Character to Target.**
- **Set the Z value of Launch Velocity to 1000.**

Test the JumpPad

- **Save your work by clicking Compile and Save.**
- **Minimize the Blueprint Editor.**
- **Use the Content Browser to place a jump pad into your level and test it.**

Double Jump

Enable Double Jump

- **In the Content Browser, open the ThirdPerson > Blueprints folder.**
- **Double-click BP_ThirdPersonCharacter to view the Character Blueprint.**
- **In the Details panel, locate the Character section.**
- **Increase Jump Max Count to the desired number (e.g., "2" for a double jump).**
- **Click Compile and Save.**
- **Click Play and test the double jump feature.**

Collectible Coins

Create a Game Mode Blueprint

- In the Content Browser, navigate to All > Content > Third Person > Blueprints.
- Double-click BP_ThirdPersonCharacter to open the Blueprint Class.
- In the My Blueprint tab, locate the Variables section.
- Click the + button next to Variables and name it CoinCount.
- Set CoinCount's type to Integer.
- Click Compile and Save.

Build a Coin

- In the Content Browser, create a new Blueprint Class.
- Choose Actor and name it BP_Coin.
- Double-click BP_Coin to open it.
- Select the Viewport tab.
- In the Components tab, click Add and add a Sphere Basic Shape Component.

Resize the Sphere

- Use the Transform tools to adjust the sphere's size.
- Move the component upward so the coin appears to float when placed in the level.
- Add a Sphere Collision component and position it around the coin.
- Select the Sphere.
- In the Details panel, under Materials, add a Material like M_Metal_Gold to give it a shine.

Customize the Coin

- Add a Rotating Movement component.
- In the Details panel, adjust the Rotation Rate to change the direction or speed if desired.
- Add a Spot Light and position it underneath the coin for a shiny glow.
- Change the Light Color to yellow-gold and reduce the Attenuation Radius to decrease the cone size.
- Click Compile and Save.

Code the Coin

- Select the Event Graph tab in your Coin Blueprint and delete any unused nodes.
- Select the Sphere1 Collider and add an On Component Begin Overlap event.
- Drag from the Exec pin and add a Cast To BP_ThirdPersonCharacter node to connect the wires.

- **Connect the Other Actor output pin on the On Component node to the Object pin on the Cast To node.**

Collect Coins

- **Drag from the As BP_ThirdPersonCharacter (blue) pin and search for Get Coin Count.**
- **Add a Get Coin Count node and connect the wires.**
- **Drag from the Coin Count pin, search for and add an Increment Int node to increase the count by 1.**
- **Attach the Cast To Exec output pin to the Increment Int Exec input pin.**
- **Drag from the Increment Int node's Exec output pin and add a DestroyActor node. Leave the target input as self.**
- **Click Compile and Save.**

Add the Coin Object

- **Minimize the Blueprints window.**
- **In the Content Browser, click and drag the coin into your level.**
- **Click Play and collide with the coin to see it disappear.**

Collectible Item UI with Widget Blueprints

User Interface Widget

- **To show the player's coin count, you'll create a Widget Blueprint.**
- **In the Content Browser, navigate to the Blueprints folder.**
- **Right-click and select User Interface > WidgetBlueprint.**
- **Select User Widget as the class.**
- **Name it HUD_Coin.**
- **Double-click HUD_Coin to open the Widget Blueprint.**

Canvas Panel

- **In the Palette panel, search for Canvas and drag a Canvas Panel into the Designer scene.**
- **Search for Horizontal Box and add it to the Canvas.**
- **Click and drag the edges to make it larger, as this will hold text and other elements.**
- **Drag two Text widgets from the Palette panel into the top-right area, placing them side by side.**

Edit the Text

- **Select the left Text block.**
- **In the Details panel, under Content, type "Coins: " for the Text field.**
- **Select the right Text block.**
- **In the Details panel, under Content, click the Bind drop-down next to Text and select Create Binding.**

Bind the Text

- **Add a Cast To BP_ThirdPersonCharacter node and connect the Exec pins between the nodes.**
- **Drag from the Object input pin of the Cast To node and add a Get Player Character node to link to the Return Value pin.**
- **Drag from the As BP Third Person Character output pin and add a Get Coin Count node.**
- **Connect its output to the Return Node's Return Value input pin.**

Playtest

- **Click Compile and Save.**
- **Click Play to test your game.**

Initialize UI

- **In the Content Browser, select Blueprints > BP_ThirdPersonGameMode and double-click to open it.**
- **Click Open Full Blueprint Editor at the top if the Event Graph isn't open by default.**
- **Add an Event BeginPlay node and attach a Create Widget node to its output.**
- **Set the Class input drop-down to Coin_HUD.**
- **Add an Add to Viewport node to the output.**
- **Connect the Create Widget node's Return Value output pin to the Add to Viewport node's Target input pin.**
- **Compile, save, and play the game. You should see the coin counter in the top-right corner of the screen!**

Sprinting Inputs and Functions

Custom Input Actions

- **In your UE5 project, navigate to All > Content > ThirdPerson > Input > Actions.**
- **Right-click and select Input > Input Action.**
- **Name it IA_Sprint and double-click to open it.**

Edit the Triggers

- **In the IA_Sprint window, click the Add Element button next to Triggers.**
- **From the Index [0] None drop-down, select Hold.**
- **Repeat for Index [1], selecting Released this time.**
- **Save and close the window.**

Input Mapping Context

- **In the Content Browser, navigate to All > Content > ThirdPerson > Input.**
- **Double-click IMC_Default to open it.**
- **In the IMC_Default window, under Mappings, click the + Adds Action Mapping button to add a new one.**
- **Select IA_Sprint from the drop-down.**
- **Select Left Shift (or your preferred key) from the drop-down.**
- **Add another control binding and select Gamepad Right Trigger (or your preferred button).**
- **Save and close the window.**

Modify the Player's Speed

- **Navigate to All > Content > ThirdPerson > Blueprints and double-click BP_ThirdPerson to open it.**
- **In the Event Graph, right-click near the bottom-right (next to the Jump nodes), type "sprint", and select IA_Sprint to add the action event.**

Create a Function

- **In the My Blueprint tab, click the + next to Functions to add a new function.**
- **Name it SprintStart.**

Functions Editor

- **In the Components tab, click Character Movement and drag it into the graph to create a reference node.**
- **Drag from the Character Movement node's pin to add a Set Max Walk Speed node.**
- **Set the Max Walk Speed to a value higher than 500 (e.g., 700-1,000).**
- **Connect the white exec output pin from SprintStart to Set Max Walk Speed.**
- **In the My Blueprints panel, under Variables, create a new Boolean variable named IsSprinting.**
- **Drag the variable into the Event Graph and select Set IsSprinting.**
- **Connect the Set Max Walk Speed node's exec output pin to the Set node's exec input pin, and check the box for IsSprinting to set it to true.**

SprintStop

- **Create a function called SprintStop.**

- **Repeat the steps above (except creating a new variable), but set the Max Walk Speed back to the walking speed (500).**
- **Uncheck IsSprinting.**

Input Action

- **Close the function tabs and select the Event Graph tab.**
- **Add a Sprint Start node as the output for InputAction IA_Sprint node's Triggered output.**
- **Click the arrow at the bottom of the InputAction IA_Sprint node to expand the additional options.**
- **Add a Sprint Stop node as the output for InputAction Sprint node's Completed output.**
- **Click Compile and Save.**
- **Playtest your game and press Shift to sprint.**

Module 4 Goals

- **Dive into game design theory and plan out your project.**
- **Create a Game Design Document to solidify your ideas for the day.**
- **Design a level or build a world for your final project.**
- **Graybox the level using Brushes in Unreal.**
- **Explore optional lessons and finalize your game level.**

Game Design Document

THEME

1. **Make a copy of the Game Design Document.**
2. **Fill in the details: Write down the name of your game, your name, and the genre you're creating.**

Theme Exploration

1. **Open this game in a new tab and play through it.**
2. **Discuss with your classmates what you think the theme of the game is.**

Decide Your Theme

1. **List out seven or eight themes for your game.**

2. **Highlight or circle your favorite ideas from the list.**

Reference Images

1. **Search online for reference images related to your chosen theme.**
2. **Create a folder on your desktop to store these images for quick access.**

Scope and Scheduling

Scope Mechanics

1. **Open your Game Design Document and navigate to the "Scope Rating" section.**
2. **Fill out the Scope Rating for your game's mechanics, considering the complexity and time required.**

Objectives and Rules

Win Conditions

1. **Consider the following common win conditions:**
    - **Achieve a position: Reach a specific location or stay there for a set time.**
    - **Wipeout/destroy something: Defeat an enemy to win.**
    - **Accumulate: Collect a certain number of items or gain the most of a particular resource.**

Lose Conditions

1. **Think about possible lose conditions, such as:**
    - **The player running out of lives.**
    - **Other players winning the game.**
    - **No other outcomes available for the player to continue.**
    - **The player exhausting all resources.**

Game Description and Pitch

Game Description

1. **Open your Game Design Document and find the "Game Description" section.**
2. **Write a brief description of your game, summarizing its core concept and gameplay.**

Pitch the Game

1. **Locate the "Hollywood Pitch" section of the document.**
2. **Craft a Hollywood-style pitch for your game, capturing its essence in a short and engaging statement.**

Level Design

Pre-Planning

Before you start designing your level, consider the following:

1. **Who is your audience?**
2. **How much time do you have to develop the level?**
3. **What is the main objective of the level?**
    - **Is it a multiplayer Capture the Flag game or a single-player quest?**
4. **How long should the level be?**
5. **What is the theme of the level?**
6. **How difficult should it be?**
7. **What makes the level unique and memorable?**

Graybox the Level

1. **Use Unreal's brushes to start laying down the groundwork for your level.**
2. **Think about the flow of each obstacle and challenge—Is it fun for the player? Does it make sense?**
3. **Refer to your sketched level draft as you build the level in Unreal to ensure alignment with your original vision.**

Blueprint Mechanics

Plan Mechanics

1. **In your Game Design Document, add or expand a section titled "Mechanics: Planning".**
2. **Write down conditions that will trigger specific mechanics (e.g., colliding with an enemy triggers a lose condition).**
3. **Detail the outcome of the mechanic being triggered (e.g., the player respawns or the level restarts for a lose condition).**

Debug Level

1. **Open your UE5 project and create a new Basic level.**
2. **Save the level as "DebugLevel".**

Create Mechanics

Win Mechanic

1. **Create an Actor Blueprint named BP_Win with a Box Collision, Static Mesh, and a custom Material.**
2. **In the Event Graph, create Blueprints that check if BP_ThirdPersonCharacter overlaps with this actor; if they do, print "You Win!".**
3. **Compile and Save.**
4. **Add the BP_Win Blueprint to the level and test it to ensure it works.**

Lose Mechanic

1. **Create an Actor Blueprint named BP_Checkpoint with a Box Collision, Static Mesh, and a custom Material, then add it to the level.**
2. **In the BP_ThirdPersonCharacter Blueprint, add a Vector variable named Checkpoint.**
3. **Create an Actor Blueprint named BP_Enemy with a Box Collision, Static Mesh, and a custom Material.**
4. **In the Event Graph, create Blueprints that check if BP_ThirdPersonCharacter overlaps with this actor. If they do, print "Game Over" and respawn them at the checkpoint.**
5. **In the Level Blueprint Event Graph, add a Character Reference and set it to BP_ThirdPersonCharacter. Set the Checkpoint variable to the BP_Checkpoint location.**
6. **Add the BP_Enemy Blueprint to the level and test to ensure it works.**

Transfer Mechanics Over

- **Compile and Save all your Blueprints and levels.**
- **Open your main level and add the Blueprints you've created (e.g., BP_Win, BP_Spawn, BP_Enemy, etc.).**
- **Copy any nodes from the "DebugLevel" Level Blueprint to your main level's Level Blueprint, updating actor references as needed.**
- **Playtest the level to ensure all mechanics are working as intended.**