

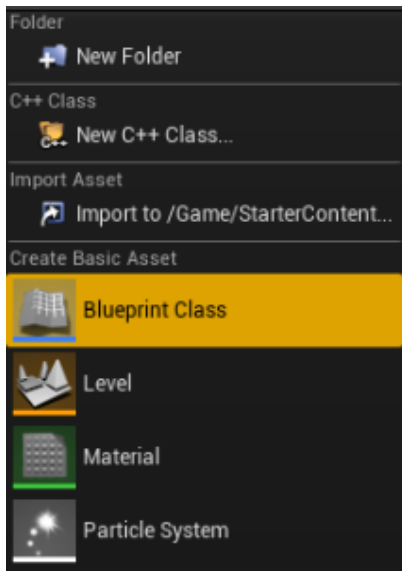
# Basic Interactivity:

## Turning on a light using Blueprints

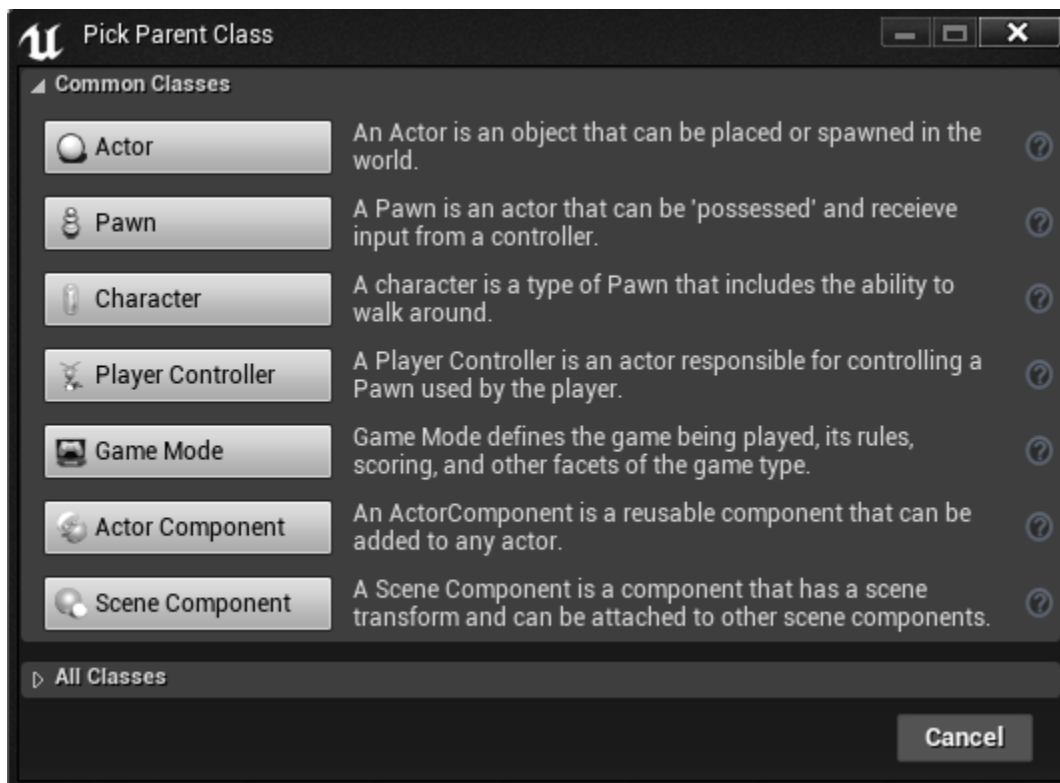
- Before we start working on Blueprints we need to open the blueprint map, navigate to the map at:  
Content/FirstPersonBp/Blueprint\_Tutorial\_Material/BluePrint\_Tutorial\_map  
and open the map.

### 1. Creating and Opening a Blueprint.

- Navigate to the blueprints tutorial assets folder, it is a blue folder the full path is:  
Content/FirstPersonBP/Blueprint\_Tutorial\_Material
- **Right click inside of the content browser** and in the menu that pops up **select Blueprint Class**.

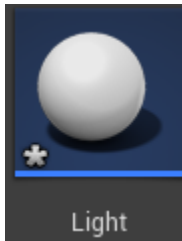


- In the menu that pops in the middle of the screen, select **Actor**.

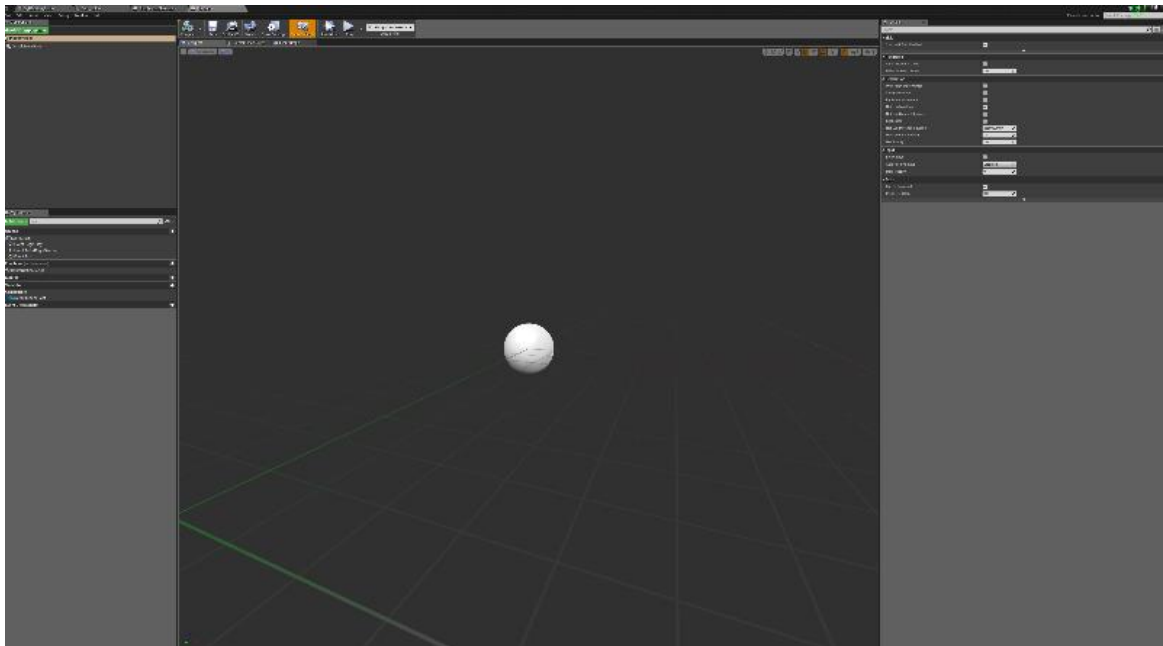


- Now that a new blueprint has been created, **don't click anything yet** the blueprint is now able to be given a name, **type in "Light"** to name it. If you have already clicked before naming it go to

the blueprint and right click on it and select rename and name it “Light”. This is what the icon should look like



- Double click the blueprint icon to open the blueprint editor the window that has popped up should look like this.



## 2. Creating a Lamp and Light

- To create a light **go to the green add component** button at the top left of the window in the components tab. In the menu that drops down **type in “Point Light”** and **select the point light option from the menu** by clicking or pressing enter on the item, do not press anything else after this. If you have not clicked anything

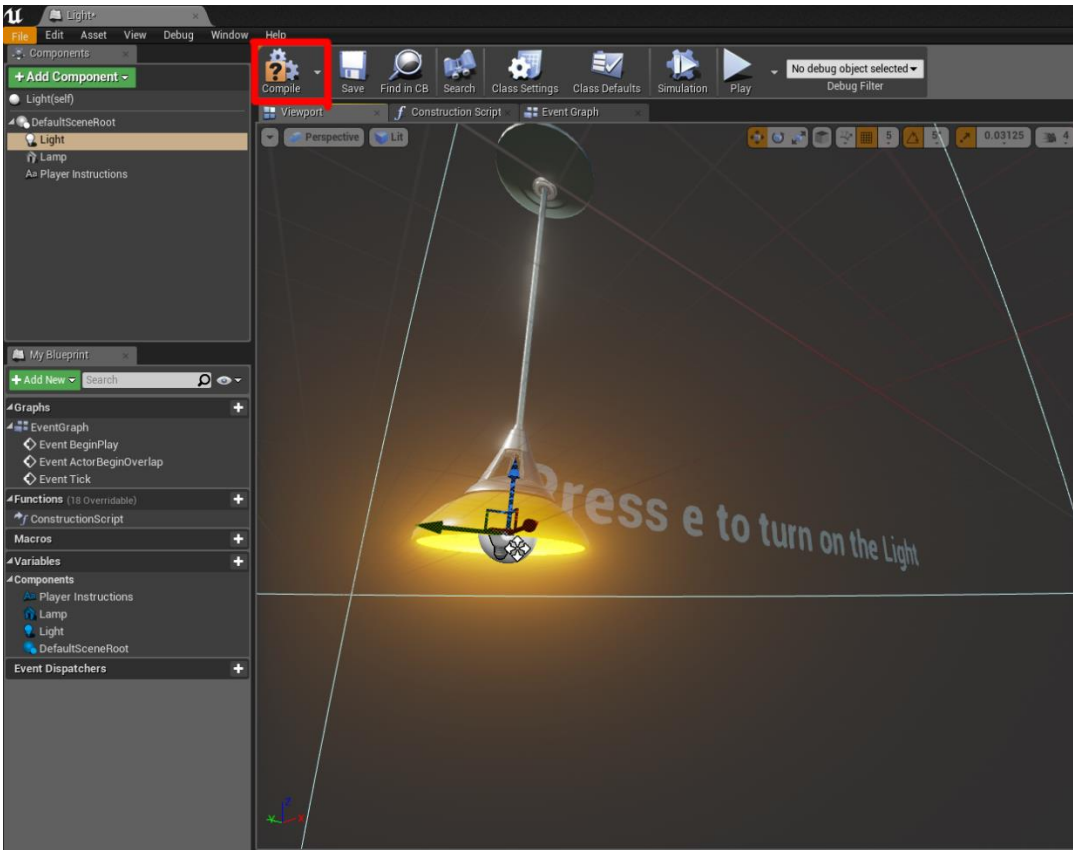
else, **name it “Light”** to name the PointLight, if you accidentally clicked the mouse just simply right click on the component that says “PointLight” and on the menu that pops up select rename and type in “Light”.

- Now go back to the content browser in the main window and in the Blueprint\_Tutorial\_Material folder **left click on the static mesh SM\_lamp\_Ceiling**, and while holding the left mouse button drag the lamp into the blueprint editor’s viewport and let go then, before clicking anything **name the mesh “Light”** .
- **Make sure that the lamp is selected** and in the details panel under the transform tab **in the Location option type in “130”** for the z axis value.



- Now that a light has been created we need to add text so the player will know what to do, **go to the green add component button** and in the menu that drops down, **type in “Text Render”** select it and **name it “Player Instructions”**.
- Now go over to the details panel and under the Text tab , **in the entry box next to Text type in “Press e to turn on the light”**

- Now hit the button that says “Compile” at the top right of the blueprints editor that



**Note:** you must always hit compile whenever you have made a change to the blueprint and want to see the change in game.

### 3. Adding the Blueprint to the Scene

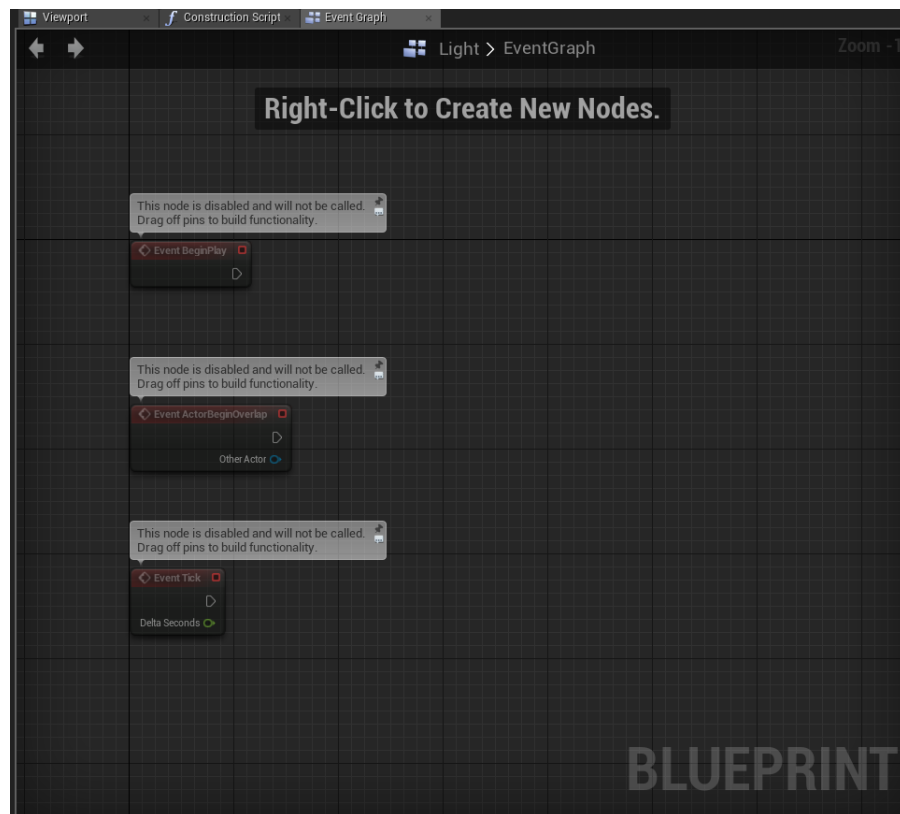
- Go to the content browser and then navigate to the Light blueprint that you have created and drag and drop it into the scene by left clicking on the light blueprint and while holding the left mouse button drag it into the scene and then let go. The blueprint and the light should now be visible in the scene.

### 4. Adding Interactivity to the Blueprint

- Go back to the blueprint editor, and at top of the viewport click on the tab that says event graph.

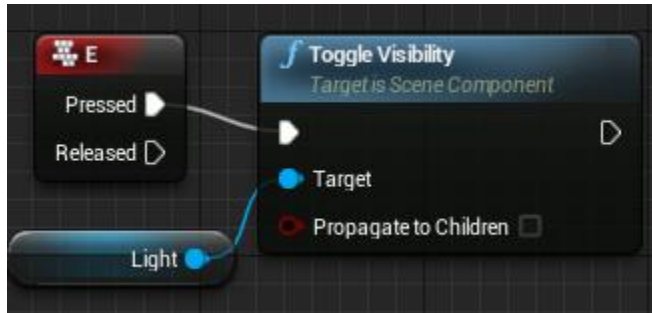


**Note:** There is some stuff already created, these are called Nodes they are the building blocks of blueprints they will contain most of the operations and logic that is carried out in blueprints, they are connected at their pins with things called wires here is what the event graph looks like.



- Right click anywhere on the gridded area except for the nodes (the boxes with red at top) and a menu should pop up, type in "event key e" and press enter or click on the highlighted item, make sure the node that pops up says e in the red area.
- On the white arrow pin on the event key "e" that has the text "Pressed" next to it and left click and while holding the left mouse button drag to the right and let go of the left mouse button, a menu should pop up that looks just like the one in the last step

in that type in “Toggle Visibility” and make sure to select the option that says “Toggle Visibility(Light)” press enter or click on it to select it and create it.



**Note:** a small blue node was automatically created this is a variable that references the light actor.

## 5. Compiling and Testing the Blueprint

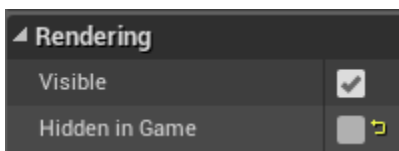
- Hit the **compile** button and then go to the main viewport and select the light blueprint by pressing the sphere icon if it is not already selected.
- In the details panel that pops up, it should be on the bottom right, go to the input tab and click on the drop down menu next to auto receive input item and select “Player 0”, This is just a quick fix to allow input we will go over doing this through the blueprint soon.
- Now hit the play button and look at the light it should be on initially now initially press the “e” key and the light should turn off press “e” again and the light should turn off.

## 6. PART 2: Adding a Trigger Box

- Now that we have the light turning on and off we now want to make sure that the player will only be able to turn the light on when he or she are within a certain area next to the light to do this **we need a trigger box**.
- **Go back to the blueprint editor and make sure that the viewport tab is selected** not the event graph editor.
- To add the trigger box **go to the green add component tab**, click on it, and in the menu that pops up **type in “Box Collision” and select it and name it “Trigger Box”**.
- Under the details panel for the Trigger Box go to the transform tab and **for the scale type in “8” for the X,Y, and Z axis**.



- This is just for debugging purposes but in the details panel **under the rendering tab, uncheck the “Hidden in Game” checkbox**, this will make the trigger box visible in game, you can turn this off later.



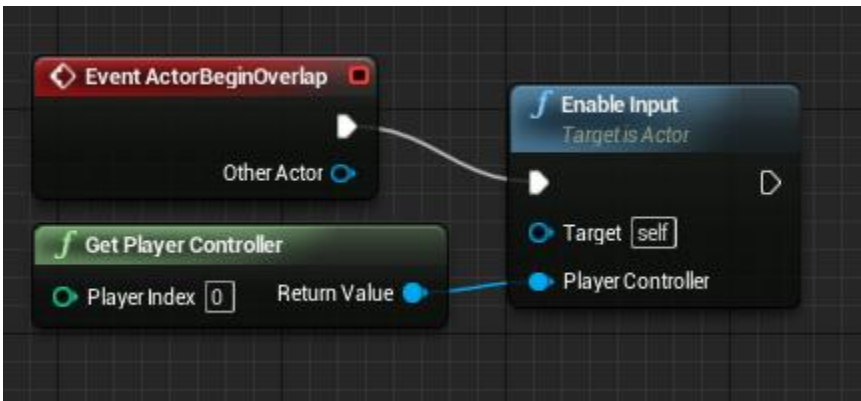
- Now **go to the event graph** in the blueprint editor.

## 7. Enabling and Disabling Input

- Before we get into the main part of this trigger box programming **we need to enable and disable the input whenever the player enters and exits the area of the trigger box**.
- Right click near the Event ActorBeginOverlap node on the grid and type in **“get player Controller”** and select it and add it to the grid



- Right click near the **get player controller node** and in the menu that pops up type in “**Enable input**” and create the node.
- Now left click on the return value pin of the **get player controller node** and while holding the left mouse button **drag the wire over to the Player Controller pin on the enable input node** and let go to connect them.
- Now left click on the **white arrow pin on the Event ActorBeginOverlap** and while holding drag the wire over to the **white arrow pin on the “Enable Input”** this will connect the two nodes.



- Now we need to **disable the input** when the player leaves the Trigger Box.
- Right click near the **event begin overlap node** and in the menu that pops up, type in “**event ActorEndOverlap**” and create the node.
- Drag off of the **white arrow pin on the Event ActorEndOverlap node** onto the grid and let go type in “**Disable Input**” and create the node.
- Now go back to the **Get Player Controller Node** and click on it, after this press “**CTRL**” and “**W**” at the same time to duplicate the

node, now click on it and **drag it over near the Disable Input node.**

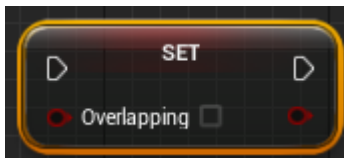
- Now **connect the new Get Player Controller node to the Disable Input node** by left clicking on the Return Value pin in the get Player Controller node and dragging the wire over to the Player Controller Pin in the Disable Input node.

## 8. Programming The Trigger Box

- In the My Blueprint panel (the bottom left panel) **click on the green “add new” drop down menu and click variable** This should automatically create a Boolean value, which is red, name it Overlapping. If it isn't a Boolean go to the details panel and in the variables tab select the drop down menu next to Variable Type and select Boolean.



- Go over to the my blueprint panel and under the variables tab **click and drag the overlapping variable into the grid and let go**, a small menu will pop up asking to get or set, **select set**. A node will pop it will look like this.



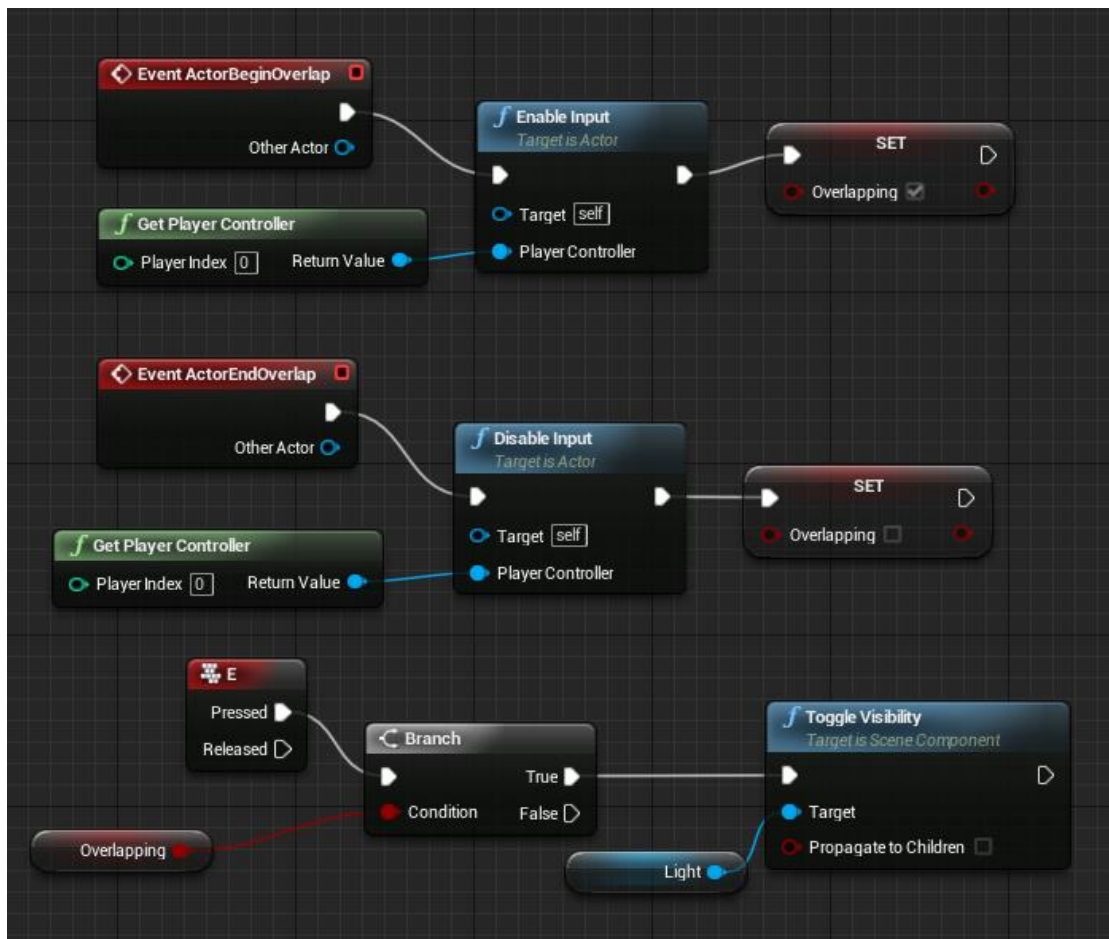
- In the set Overlapping node, next to the text that says overlapping there **is an empty check box click it to add a check** this is setting the Boolean to true.
- **Now connect the Set Overlapping node to the Enable Input node** by clicking on the white arrow pin on the Enable Input pin and

dragging the wire to the white arrow pin on the Set Overlapping node, this will connect them.

- Now the blueprint needs to be notified once the player has left the trigger box we have created so we need to create an actor end overlap node
- Find the “Event ActorEndOverlap”.
- Now go back to the set overlapping node from earlier steps and click on it and hit “CTRL” and “W” at the same time to duplicate the node and then drag it over near the Disable Input node.
- Go to the check box in the NEWLY created set node and click it to get rid of the check in the box this is equivalent of setting the Overlapping variable to false.
- Now connect the disable input node and the new set overlapping node by clicking on the white arrow pin on the Disable Input node and drag the wire to the NEW set node’s white arrow pin creating a connecting wire.
- Now go over to the event e key node right click on the pressed pin and in the menu that pops up select “break links to Toggle Visibility”.
- Now from the event key e nodes pressed pin click and drag to the left and let go, in the menu that pops up type in “branch” and create the node, a branch node should pop up, this is the equivalent of an if statement in programming.
- Now go back to the variable tab on the blueprint panel and click and drag the overlapping variable into the grid behind the branch

node and let go a menu will pop up that asks to get or set, select get.

- Click and Drag from the overlapping's get variable pin and drag the wire into the "Condition" pin on the bottom left of the branch this connects the variable with the branch.
- Now click and drag from the True pin on the branch node and connect the wire into the toggle visibility node's white arrow pin this connects the two nodes together, this is what the completed blueprint should look like.



## 9. Testing Out the Blueprint

- Now hit the compile button.

- Now go back to the scene and hit play, now try to turn the light off from outside of the trigger box and it won't work, now go into trigger box area and try pressing, it should work.

## APPLICATION TIME

- Change blueprint so the light can be turned on and off from very far away.

### Challenge

- Change it so the player must hit "space bar" to turn the light on and off.

# Making a Moving Platform


## 1. Creating The Blueprint

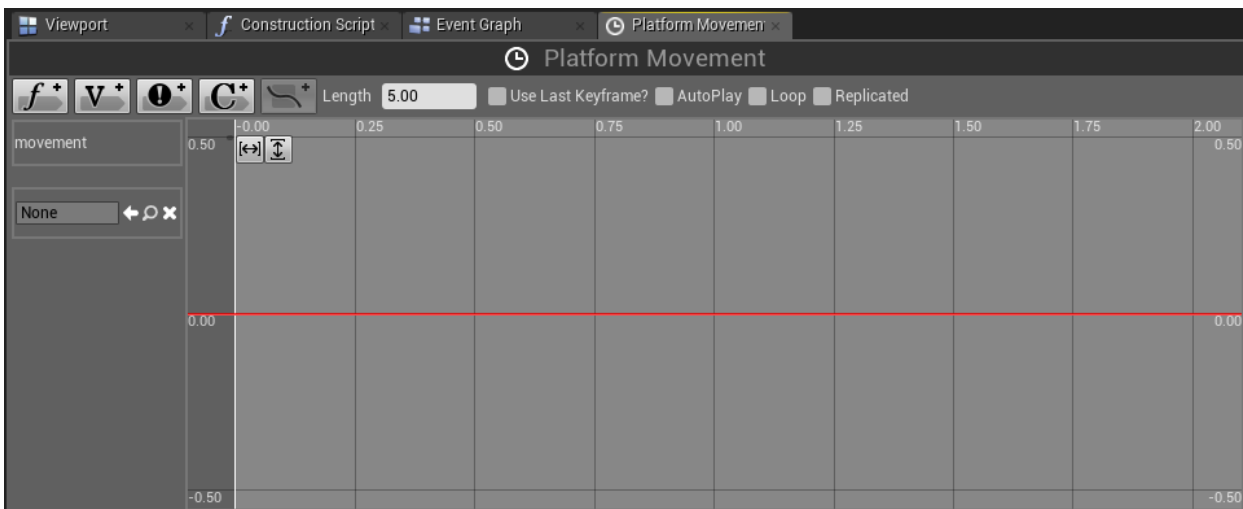
- Right click in the content browser and select "Blueprint Class" from the menu that pops up.
- Select "Actor" from the Pick Parent Class menu, This will create a new blueprint Class, once it is created name the blueprint "Moving Platform" and the
- Double click the Moving Platform blueprint to open the blueprint for editing.

## 2. Setup

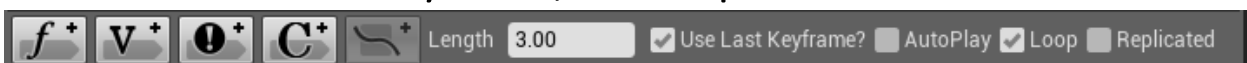
- Now we will add the platform to the scene, Go to the content browser in the main editor window, go to the Blueprint\_Tutorial\_Material folder and **click on the platform and drag it into the blueprint viewport**. It will automatically assign the name to the static mesh component.
- Now to start creating the blueprint **go to the “Event Graph”** which is located at the top of the viewport of the editor.

### 3. Creating Movement

- Drag of the pin of the Event Tick node and type in “add Timeline” and create the node.
- Name is “Platform Movement”
- Double click the timeline node and you will open the timeline editor.
- Click the button at the top of the timeline editor window that looks like this  and name it “movement”, the timeline editor should look like this.



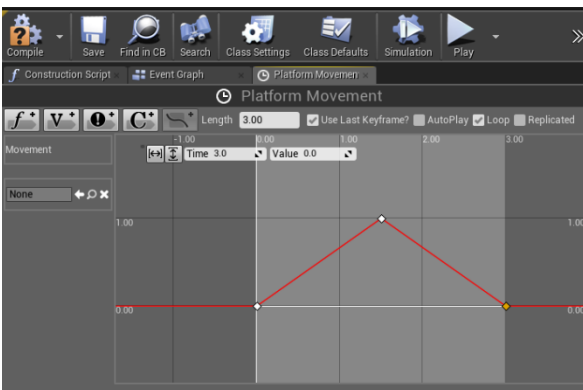
- Now on the bar above the timeline graph **change the length to 3.00, and check Use Last Keyframe, and loop**



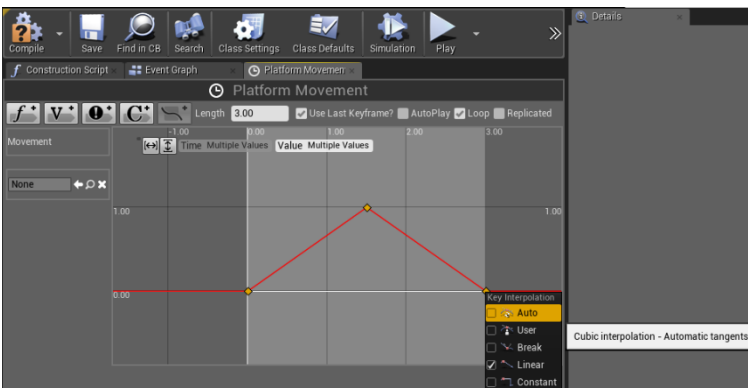
- Now anywhere on the red time track Press “Shift” and left click at the same time then once the point has been created type “0” into both time and value input boxes that pop up near the top of the timeline editor



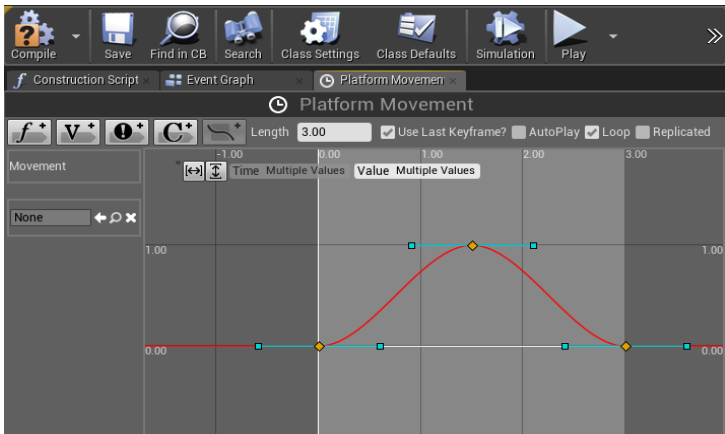
- Create two more points by shift clicking on the red line, in the second point type in the value input boxes that pop up “time = 1.5” and “value = 1” and then for the third point type in “time = 3” and then “value = 0” it should look like this.



- Now press “CTRL” and click all three of the points at the same time, then right click on any of them and select auto from the menu that pops up
  - You must click on the yellow points, NOT in empty space.



- The graph should now look like this.



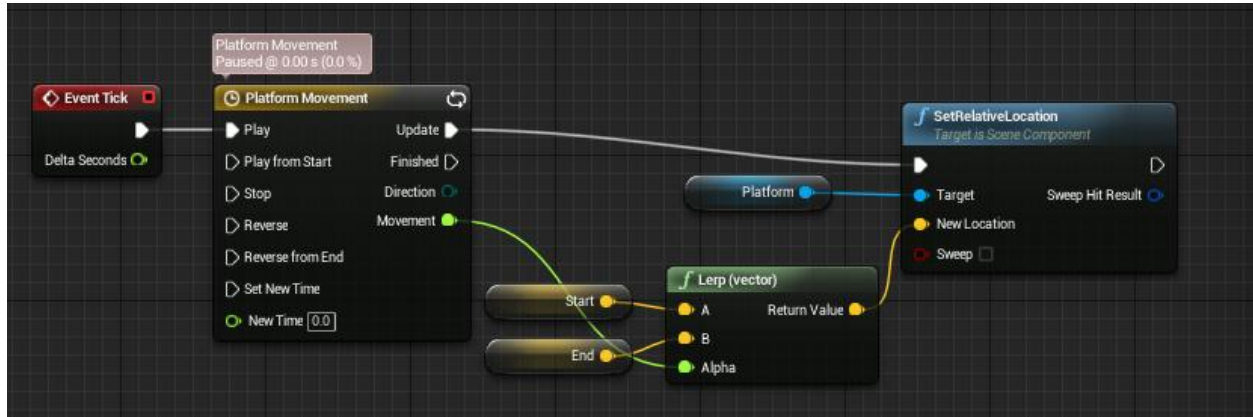
- Now **go back to the event graph** by clicking on the event graph tab at the top of the timeline editor.
- On the **“Update”** pin on the right of the **“Platform Movement”** timeline node tab **click and drag the pin to the right and type in “set Relative location”** and **make sure to select Set Relative Location(platform)** from the options that pop up.
- Now go back to the timeline node and **click and drag off of the green (float) movement pin** and let go then in the menu that pops up **type in “lerp”** make sure to select **lerp (vector)** this is a very important part.
- Now go to the top left yellow pin title A on the lerp node right click on it and in the menu that pops up select **promote to variable name** it **“Start”** .
- For the B pin in the Y value type in **“300”**.



- Then **right click on the B pin** and select **promote to variable** in the menu that pops up **name it “End”**.
- **Now click and drag from the right of the lerp(vector) node** and drag it on to the **new location pin of the Set Relative Location pin** and let go to connect the two nodes.



- Here is the finished blueprint.



## 4. Testing the blueprint

- Now go to the content browser and drag the blueprint into the scene.
- Hit play and the platform should move back and forward on the y axis.

## APPLICATION TIME

- Make the platform go further in either direction by changing the Y variable of “end” from 300 to a larger number.

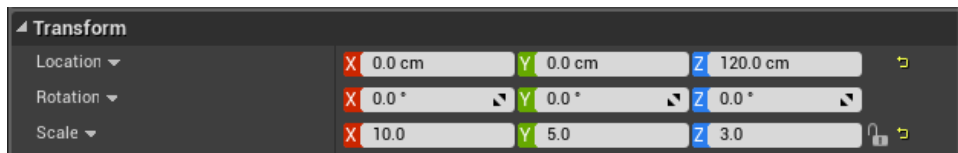
## Challenge

- Make the platform go up and down instead of left and right.
  - OR
- Make the platform move slower.

# Opening door

## 1. Setting up the Door Blueprint

- To start this off we will **create a new blueprint** go to the content browser and right click in the content browser and select create new blueprint **make it an actor** in the menu that pops up **and name the blueprint “Opening\_Door”**.
- Open the blueprint and then go to the content browser to the folder Blueprint\_Tutorial\_Material folder and **drag and drop the static mesh named “Frame”** into the blueprint editors viewport, it should automatically name the component.
- Repeat this step but with the static mesh door and name it **“Door”** for later convenience.
  - You may need to line the door up so it is inside of the frame.
- Go to the add component drop down menu click on it and type in **“Box Collision”** and **Name it “Trigger Box”** go to the in the details panel under the transform tab type in for the Z location type in **“120”** and for scale **“10”** for X ,**“5”** for Y ,and **“3”** for Z




## 2. Making the door rotate

- Switch to the event graph editor in blueprints
- Just like the light earlier we need to make sure that the player is within range of the door to open it.
- Right click on the grid in the menu that pops up type in **“event actor end overlap”** and create it.
- In the My Blueprint panel, Go to the add new drop down menu and click it, select to create a new variable, name the variable

Overlapping and make sure it is a Boolean, if it is not go to the details panel and next to variable type from the drop down menu select Boolean.

- Now drag and drop the Overlapping variable into the scene let go and select set from the menu that pops up.
- Click on the set Overlapping node then Hit “CTRL” and “W” at the same time to duplicate the node, drag it below the original set Overlapping node.
- In the original Set Overlapping node click the check box to make it true and then connect the event ActorBeginOverlap to the Overlapping set node by clicking on the white arrow pin and dragging the wire to the set Overlapping node, which will make the variable true
- Now drag off of that Set Overlapping node(true) and let go in the menu that pops up type in “enable input” and create the node.
- Then connect the event ActorEndOverlap to the duplicated set Overlapping node that doesn’t have a check in it(False)
- Then drag off of that Set Overlapping Node(False) and in the menu that pops up type in disable input.
- Now right click in between these two small networks and in the menu that pops up type in “Get Player Controller”, then connect it to the player controller pins on both the enable and disable input nodes.
- Right click on the grid and in the menu that pops up type in “event key e” and create the node

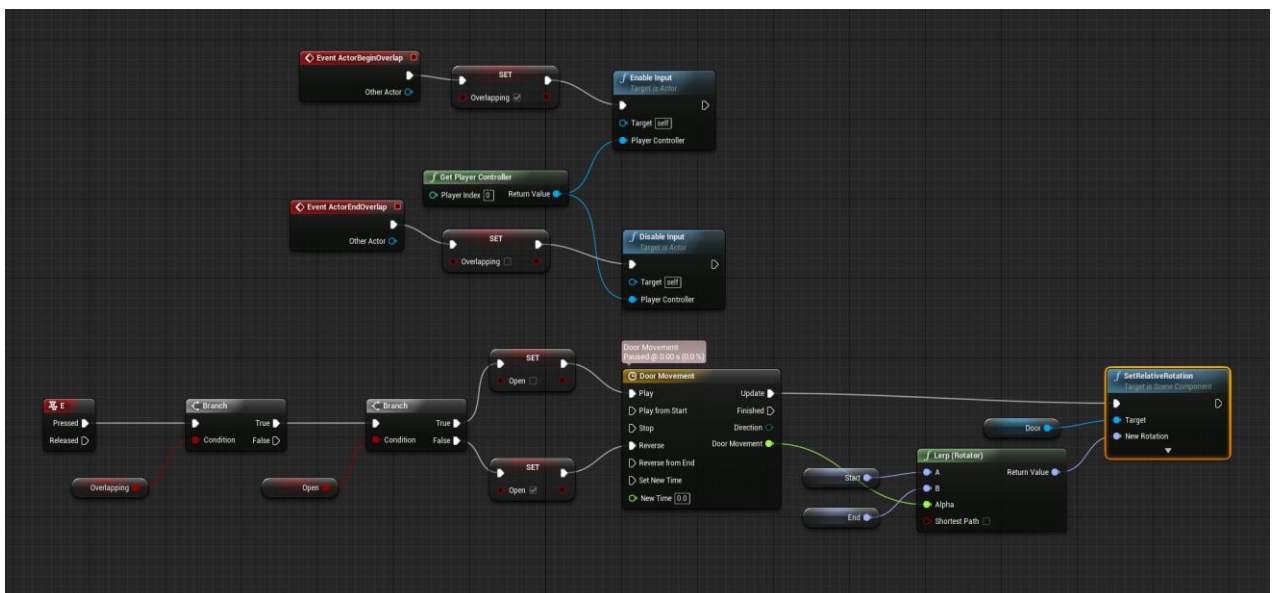
- Click and Drag off of the pressed pin on the event “e” node and let go in the menu that pops up type in “Branch” and create the node
- Now drag an Overlapping variable into the scene let go and from the menu that pops up select get connect then get Overlapping pin into the condition pin on the branch by left clicking on the Overlapping pin and dragging the wire into the condition pin of the branch.
- Now off of the true pin on the branch node drag off of it and in the menu that pops up type in “Branch” and create the node.
- Now we need to Create another Boolean variable, go to the my blueprints panel and click on the Add New Drop down menu select variable from the menu that pops up name the variable “Open”, HIT COMPILE then go to the details panel and under the default value click the check box to make the initial value true.
- Now drag the open Boolean onto the grid and let go and a menu will pop up asking get or set, select get from the menu that pops up.
- Click and drag the get open variable’s pin and drag the wire into the condition pin on the second branch.
- Now go to the variables tab and drag the open variable onto the grid and let go and from the menu that pops up select set.
- Now click on the set Open node and Hit “CTRL” and “W” at the same time to duplicate the node.
- Connect the True pin from the second Branch to the original Set Open’s pin

- On the duplicated set Open node check the box setting the variable to true.
- Connect the false pin from the second branch to the second (true) Set Open Pin.
- Now we need to add a timeline for the movement.
- Right click on grid and in the menu that pops up type in “add timeline” and create it and name the timeline “Door\_Open”
- Double click on the timeline to open the timeline editor
- Hit  at the top of the timeline editor to create a new function track, name it “Movement”
- Now change length(located at the top) to “2.00” and check use last key frame.



- Now shift left click near the beginning of the red line and type in in 0 for time and 0 for the value when the options pops up once the point is located, like earlier.
- further down the red line press “Shift” and left click at the same time and in the value boxes that pop up at the top type in for time “2” and then for value type in “1”.
- Then “CTRL” and left click both of the points and then right click on either one of them and from the menu that pops up hit auto to smooth out the movement.
- Now go back to the event graph.
- Connect the “set open” node with no check(false) to the play pin on the time line.

- Then connect the Set Open node with the check(true) to the reverse pin of the timeline
- Drag from the movement pin on the right side of the door movement timeline and let go in the menu that pops up type in “lerp rotator” and create the node.
- Right click on the A pin and in the menu that pops up click promote to variable name it “Start”.
- Then in the B values type in 90 for y (y is yaw in this case not the y axis) then right click on the B pin and in the menu that pops up hit promote to variable name it “End”.
- Now drag off of the Return Value of the lerp node and let go, in the menu that pops up type in “set relative rotation” (make sure to select the one with “door” in parenthesis) make sure it is only rotation, not rotation and location.
- Now connect the door open timeline from the update pin and connect it to the set relative rotation node.
- This is what the network should look like.



### 3. Testing out the door

- Hit compile
- Drag the door Opening blueprint into the scene
- Hit play and press “e” to open the door then press “e” to close the door, it should only work in the trigger box we have set up

### APPLICATION TIME

- Make the door open from the bottom so that it looks like a draw bridge.

### Challenge

- Make the door open very slowly at first then suddenly burst open.

# Lever and door system

## 1. Notes

- Now we will create a lever and door system now go to the area in the map that says **button and door system** there will be a pre made blueprint there, To open the blueprint select the **Button\_BP** in the blueprint example map and in the details panel click **edit blueprint**, there will already be a couple of steps done, that you have completed earlier

- There will be an “event E” node that is connected to a branch that will check if the player is within the range of being able to press the button, there will also be an enable and disable input system set up

## 2. Creating a Button

- We want to add a little sign to the player that he or she has hit the button so we will create a very small visual to show that they have hit the button
- Before we get to setting the location of the button we need to create a few new variables to allow the button to be reset, go to the my blueprint panel and click the add new menu and **create a new variable name it “Pressed” make sure that the variable is a bool it should be red**, if it is not go to the details panel and click the variable type drop down menu and select Boolean.
- Now we need a vector variable, go to the add new drop down menu and **create a new variable, name it “End Location”** now go to the details panel, **under the variables tab and on the variable type drop down menu click it and select vector it should be yellow.**
- **Drag off of the true pin on the branch and let go in the menu that pops up type in “Branch” to create a new branch.**
- **Go to the pressed variable and drag it into the grid let go and select get from the menu that pops up, connect it to the second branches condition node.**

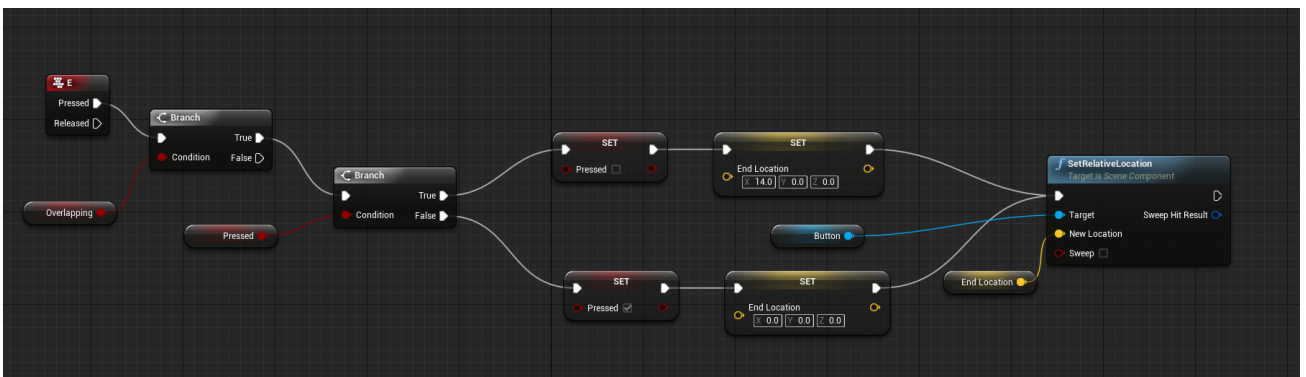


- Now go back and drag “pressed” again into the grid and let go select set from the menu that pops up.
- Click on the Set Pressed node Hit “CTRL” and “W” at the same time to duplicate the node.
- In the new set “Pressed” node click the check box to make the value true.
- Now connect the true pin from the second branch into “pressed” set node that does not have the check box checked(the original one).
- Connect the false pin from the second branch to the duplicated set “Pressed” node (the false one).
- Now go to the end location vector variable from the variables tab and drag it into the grid, let go and from the menu that pops up select set.
- Click on the set “end location” node and Hit “CTRL” and “W” to duplicate the node.
- Connect the Duplicated one to set pressed node that has the checkbox checked(the set “Pressed” coming off of the False pin on the branch it is also true).
- On the original set change the value of the x value on the node to 14 and connect the node to the set pressed that’s checkbox is not checked(the set “Pressed” coming off of the true pin of the branch it is also false).
- Right click on the grid and in the menu that pops up type in “set relative location” and select set relative location(Button) to create the node.

- Now go to the variables tab and drag and drop “endlocation” into the grid and from the menu that pops up select get.
- Connect the “end location” variable to the “new location” pin in the “set relative location” node.
- Now connect both of the set end location nodes to the set relative location white arrow pin.
- Now that there will be a small visual we need to actually have the blueprint communicate with the door blueprint that you created earlier.

### 3. Making the Blueprint Communicate

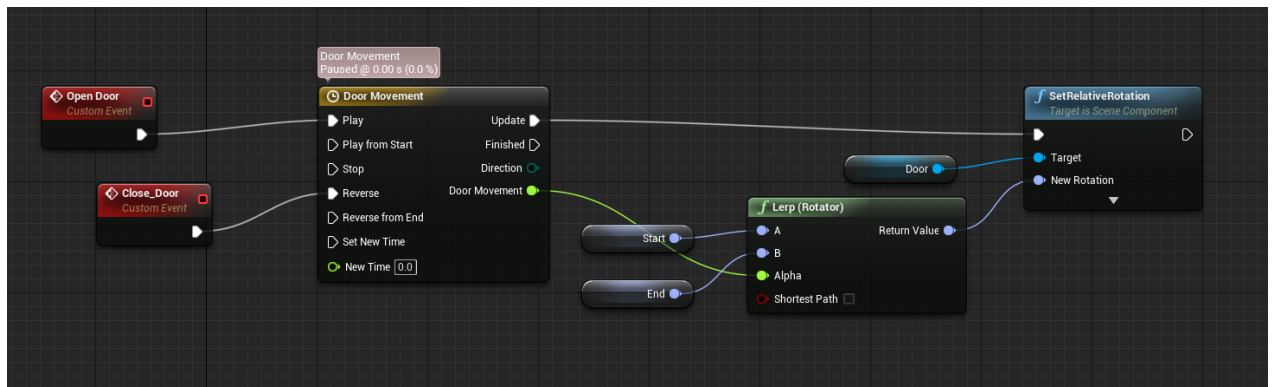
- Go to green add new drop down menu in the my blueprint panel and create a new variable, name the variable “Door Blueprint”.
- Now go to the details panel and click on the variable type drop down menu and type in “Opening\_door”, make sure to click the blue sphere object reference variable for this to work, not the purple one.





- Now that we have a reference to the door blueprint we need to slightly modify the door blueprint, go to the door blueprint in the MAIN editor window and move it over to an area that is near the

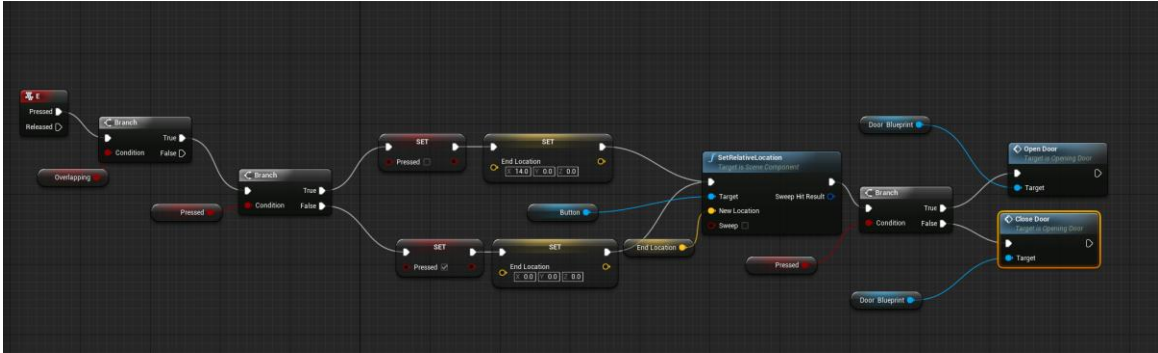
switch now go to the details in the editor panel and select edit blueprint

- Go to the event graph and then on the door\_open timeline go to the play pin and right click it and select break link to set open?().
- Do this step again but with the reverse pin and break the link.
- Now we will create two custom events
- Right click on the grid and in the menu that pops up type in “custom event” create the node and name it “Open\_Door”.
- Now repeat that step but name the event node “Close\_Door”.
- Now connect the Open Door event node to the play pin on the timeline.
- Then connect the Close Door event node to the reverse on the timeline.
- This is what it should look like.



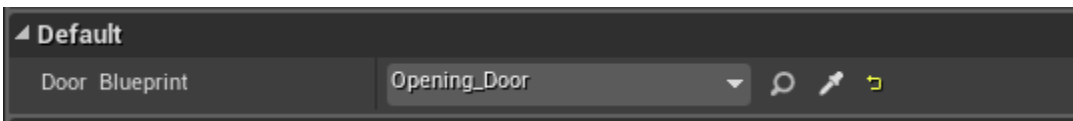
- Now that we have edited the door blueprint go back to the **Button blueprint** by going to the tab at the top of the screen that says lever or by going back to the window (if you moved the blueprint editor somewhere else on the screen)

- Now go to the My Blueprint panel and under the variables tab drag and drop the “Door Blueprint” variable into the grid and from the menu that pops up select get.
- Click on the get Door Blueprint node and Hit “CTRL” and “W” at the same time to duplicate the get Door blueprint variable.
- Now drag of the first get Door blueprint pin let go and type in “Open Door” in the menu that pops up, and create the node.
- Now go to the second get Door Blueprint and drag off of the pin and let go and in the menu that pops up type in “Close Door” and create the node.
- Now drag off of the set relative location node and let go somewhere on the grid and in the menu that pops up, type in “Branch” and create the node.
- Now drag the “pressed” variable from the variables tab into the grid and let go select Get from the menu that pops up and then connect it to the branches condition pin.
- Connect the branches True pin to the Open\_Door node
- Connect the branches False pin to the Close\_Door node
- Before this blueprint will work go to the variables tab under the my blueprints panel and click on the gray closed eye  next to the variable and it should now be an open eye  this means that this variable is public.
- This is what you are supposed to have



## 4. Testing the Blueprint in the level

- Hit the compile button
- Go to the editor and select the switch from inside of the viewport by clicking on it details panel in the default tab there is now a section called door blueprint and click on the drop down menu and there should only be one option called “Opening\_door” select that. Now the variable is assigned to a blueprint inside of the editor and can now communicate with it



- Hit play and go inside of the buttons trigger box and press “e” and the switch should depress and the door should open now press it again and the switch will reset and the door will close

## APPLICATION TIME

- Have a light turn on at the door when it opens

## Challenge

- Have a red light turn on at the button when the door is open and off when the door is closed.

# Health system

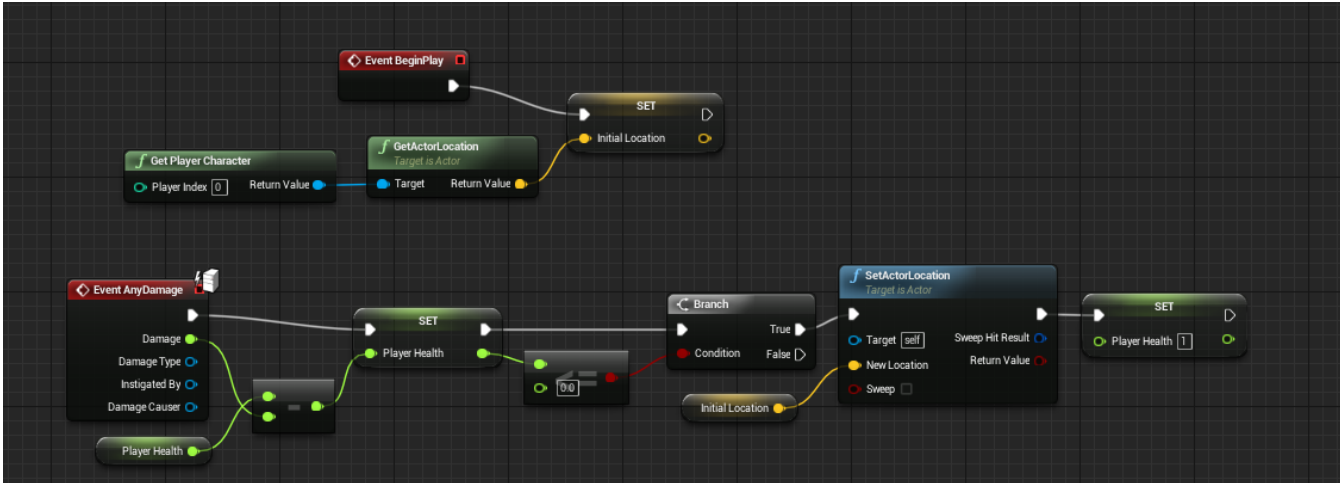
## 1. Setting up a health System

- Open up the first person character blueprint to start working on the health system, the folder is located in the project at Content/FirstPersonBP/Blueprints/FirstPersonCharacter\_1.
- First we need to get the players initial location so we can have a simple respawn.
- **Create a new variable** and **name it “initial location”** then go to the details panel and in the variable type option **change its variable type to a vector**.
- **In an empty area** on the grid **right click in the grid** and in the menu that pops up type in and **create an “event begin play” node**.
- Now **right click near the Event Begin Play node** and in the menu that pops up type in **“get player character”** and create the node.
- **Drag off of the return value pin of the Get Player Character node and let go** type in **“get actor location”** into the menu that pops up and **create the node**.
- Now **drag off of the return value of the Get Actor Location node and type in “set initial location”** and select it to **create the node**.
- Now **connect the Event Begin Play node and Set Initial Location node**.

- We now need to **create a player health variable**, go to the add new drop down menu and **add a variable name it “Player Health”** then go to the details panel and go to the variable type option and select float to **change it to a float**.
- **Hit the compile button**, and then go to the details panel and **under the default value for “player health” type in 1**, we want this to be one because the health will be between 0 and 1 for later use with a HUD.
- Find an open spot on the grid, **and right click and type in “event any damage” and create the node**.
- Now we want to be able to handle the damage that is being sent to this event node so **drag and drop player health variable into the scene** near the event any damage node let it go **select get** from the menu that pops up.
- **Now drag off of the player health variable** and let go and in the menu that pops up **type in “subtract float” and select the subtract float-float node**.
- Now that that is created **drag from the event any damage’s, green (float) damage pin and connect it to the bottom of the subtract pin**.
- Now **on the right pin of the subtract node click and drag off of it and let go** in the menu that pops up **type in “Set Player Health” and create the node**.
- **Connect the Event Any Damage node to the Set Player Health node**.

- Now from the green pin on the right of the Set Player Health node drag and let go type in “less” and select the float  $\leq$  float option to create the node.
- Now from the red pin on the right side of the less than node drag off of it and let go and in the menu that pops up type in “branch” and create the node.
- Now connect set player health and the branch.
- From the true pin drag off and let go and in the menu that pops up type in “Set Actor Location” and create the node.
- Now drag the initial location variable we created earlier into the grid and from the menu that pops up let go select get.
- Now connect the initial location variable to the new location pin on the left side of the set actor location node
- Now drag and drop the player health variable into the graph and let go, from the menu that pops up select set and create the node.
- Connect the Set Actor Location node to the Set player health node.
- Now set Actor Health Variable to 1 by clicking on the text entry box in the Set Player Health Node and type in 1.
- Here is what you should have so far.





- Now hit compile we will now move on to the item that will be damaging the player, the lava.

## 2. Creating the Lava Blueprint

- SAVE NOW. CTRL + S.
- Create a new blueprint, and in the menu that pops up select actor, name it “Lava”.
- Open the blueprint.
- Go to the content browser and navigate to content/FirstPersonBp/Blueprint\_Tutorial\_Material/Plane and drag and drop the plane static mesh into the blueprint scene.
  - This asset may need time to load so if your computer freezes or stops just give it a second.
- Now create a box collision by going to the Add Component drop down menu and typing in “Box Collision” name it “Trigger Box” then under the details panel change the transform variables to “5” for scale on the x axis and “10” for the scale on the y axis.

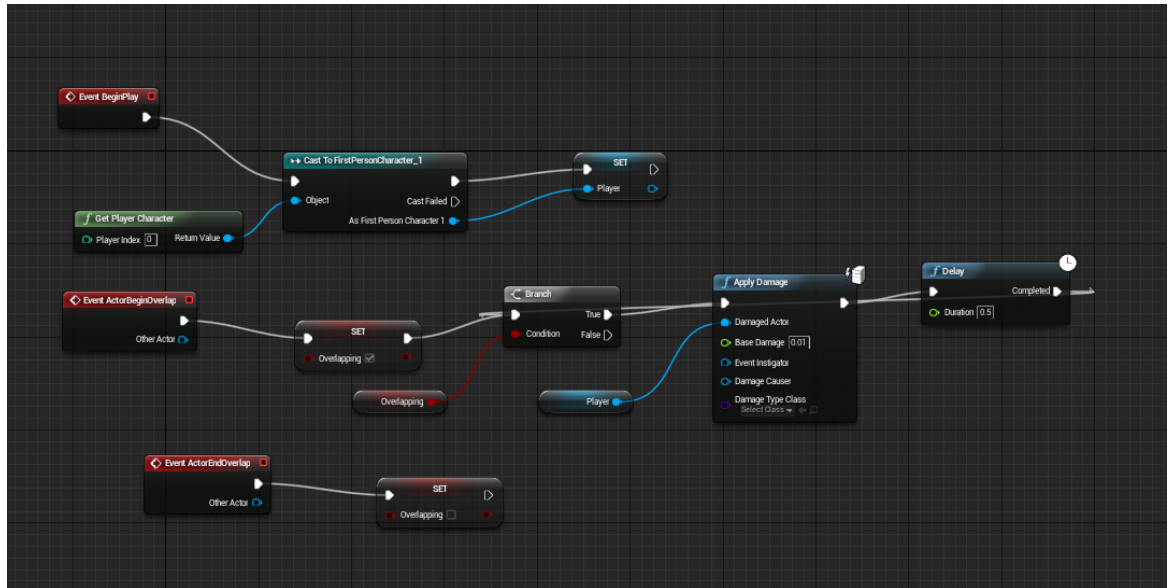


- Now switch to the event graph.

- Near the event begin play node right click on the grid and type in “Get Player Character “and create the node.
- On the Get Player Character node drag off of the return value pin and type in “cast to first person character\_1“ and create the node.
- Connect the event begin play node to the Cast To FirstPersonCharacter node.
- On the right of the “cast to first person character” right click on the “As First Person Character” pin right click on it and in the menu that pops up select promote to variable name the variable “Player”, now it will cast to the player.
- Now create a new variable, and name it “Overlapping” and make sure that it is a Boolean variable type by going to the details panel and changing it to Boolean if it is not
- Now drag off of the white execute pin of Event Actorbeginoverlap node and let go type in “set Overlapping” in the menu that pops up and hit enter now check the box in the set node to make it true.
- Now right click on the grid and in the menu that pops up type in “Event Actor End Overlap” and create the node
- Now drag off of the Event Actor End Overlap white arrow pin and in the menu that pops up type in “Set Overlapping” and create the node.
- Now go back up to the original “set player” on(off of the event begin actor overlap) and drag off of the set Overlapping node

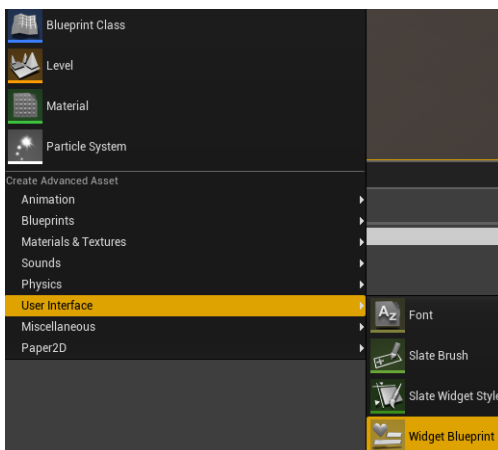
and in the menu that pops up type in “Branch” and create the node.

- Drag the Overlapping variable onto the grid from the my blueprint panel and in the menu that pops up select get.
- Now connect the get Overlapping node with the condition pin on the branch node.
- Now drag off of the true pin of the branch and let go in the menu that pops up type in “apply damage” and create the node.
- In the Base damage box on the Apply Damage node type in 0.01
- now drag the Player variable into the scene and let go in the menu that pops choose get and then connect it to the damaged actor pin on the left of the apply damage node.
- Now drag off of the left of the apply damage node and in the menu that pops up type in delay and create the node.
- Change the duration of the delay node to 0.5 seconds by clicking on the box next to duration.
- Now from the completed pin on the delay drag it backwards all the way back to the left pin on the branch and connect it there you have just create a very simple loop now hit compile and everything should be working here is what you should have.



### 3. Linking the Blueprint with the HUD using UMG

- Now we want to display this health system on a HUD.
- Go back to the main editor
- Right click in the content browser and go to user interface section in the menu that pops up and create a widget blueprint and name it HUD.



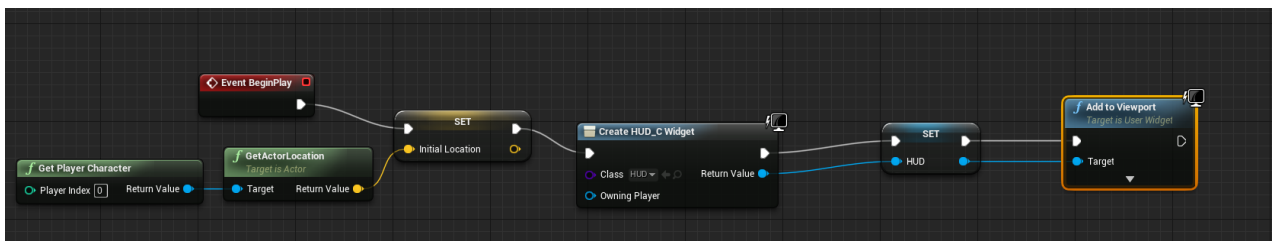
- Before we start editing the HUD we need to prep the first person character blueprint. Now go to the first person character and on the event begin play node network **drag off of the set**

initial location variable and in the menu that pops up “type in create widget” and create the node.

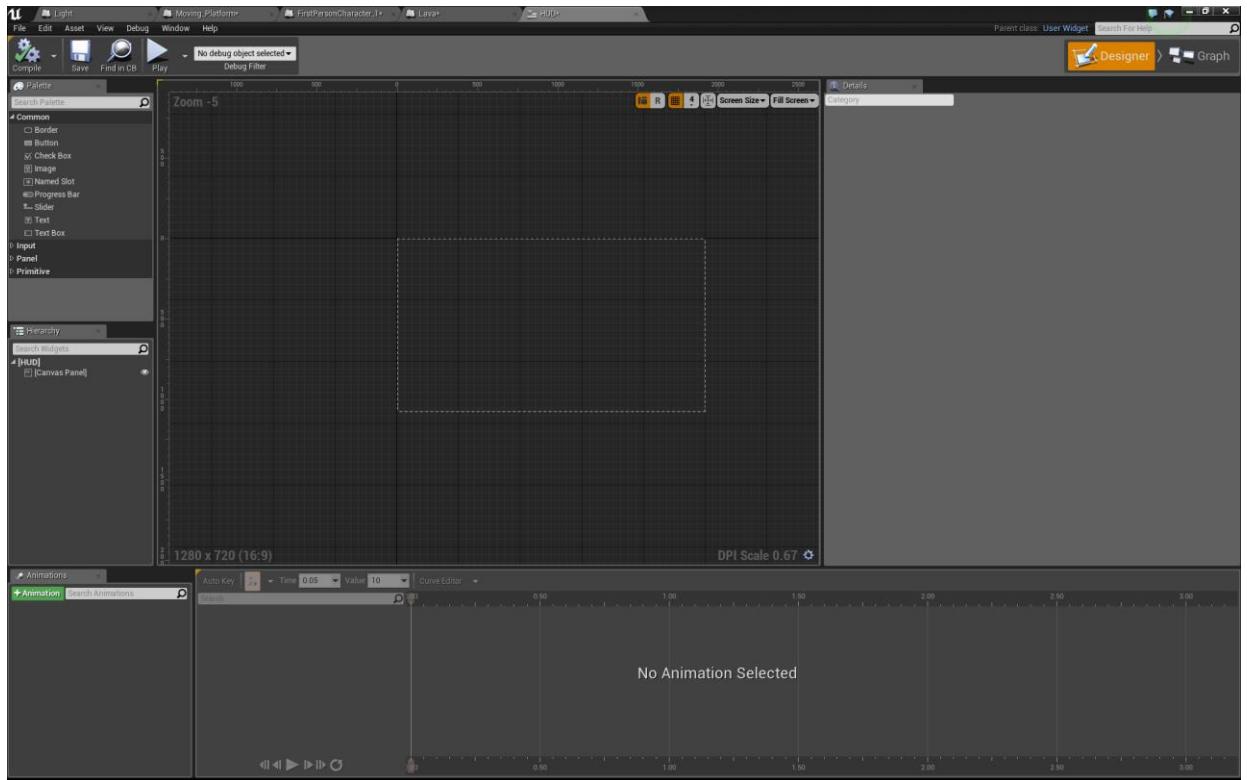
- In the purple class drop down box click it and select HUD from the options it should look like this.



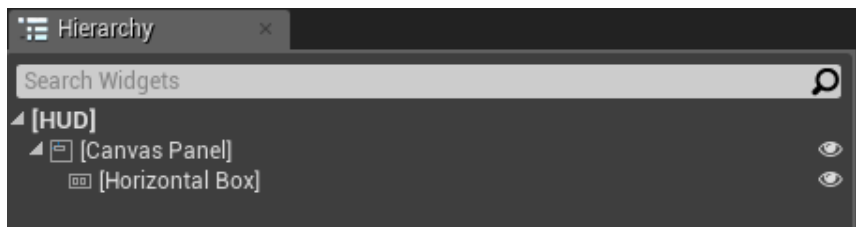
- Now right click “return value” and in the menu that pops up select promote to variable and name it “HUD”.
- Now on the Set HUD node drag off of the right blue pin and type in add to viewport and create the node
- Here is the blueprint finished.



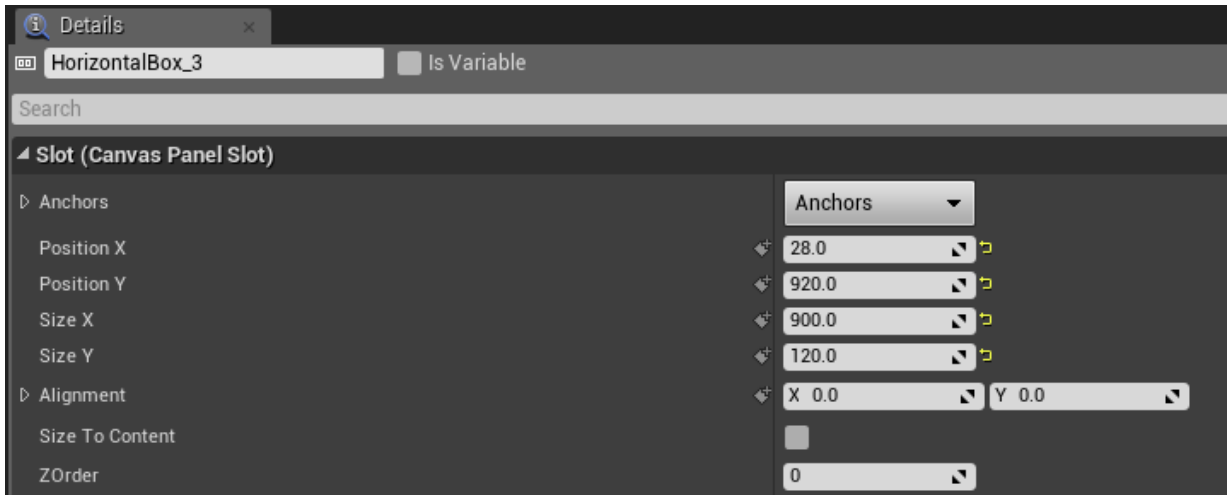
- Now go back to the content browser and Open the HUD blueprint it should look like this.



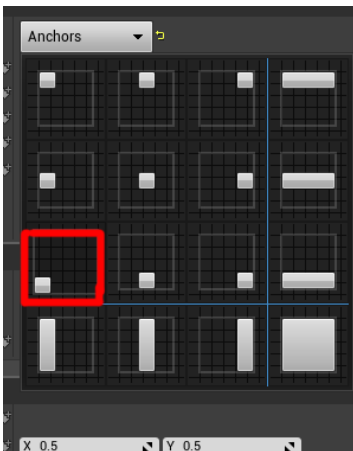
- Now we need to create a box to hold the health bar so **under** the pallet panel, drop down the “panel” tab and drag the horizontal box down to the hierarchy panel and let go while hovering over the canvas panel tab it should look like this.



- Now in the details panel for the horizontal box, under the slot tab scale the horizontal box to “28” for position x, “920” for position y “990” for the x size, and 120 for the y size.



- Under the anchors menu click on the bottom left one as shown.



- Now go back to the palette panel and expand the common tab down drag a “text” item down to on top of the horizontal box to have it be the horizontal boxes child.
- Go to details on the text item and under slot horizontal box slot change the vertical alignment to center

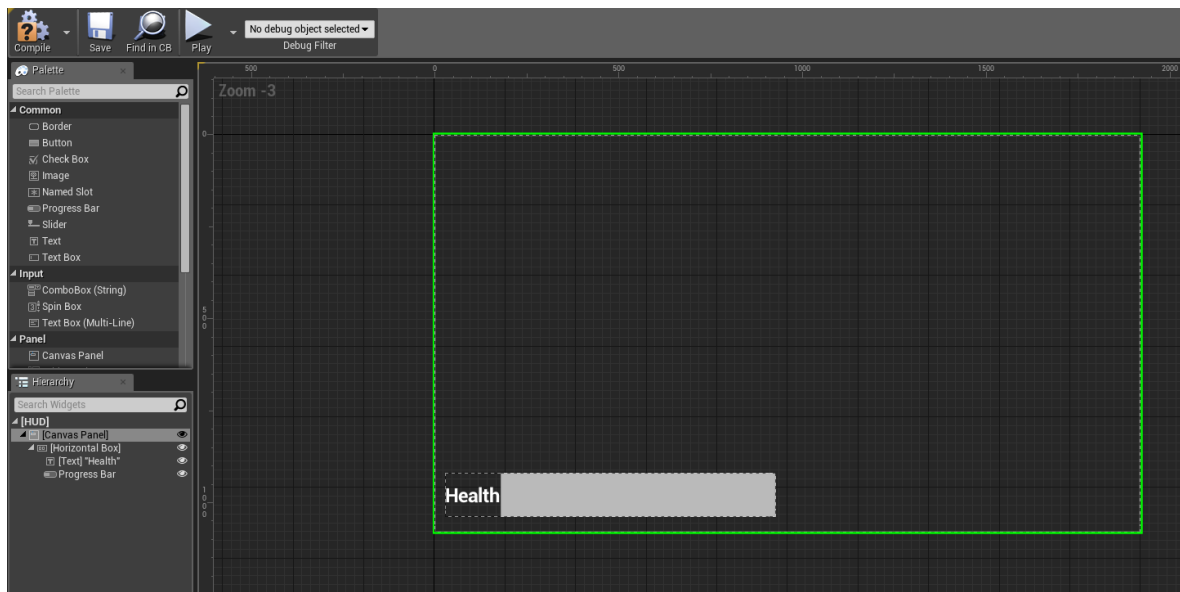


- Go to the content tab under the details panel and in the text option type in health
- In the details panel Go to the appearance of the text and in font change the size to 40

- Drag a progress bar from the common tab and drag it onto the horizontal box.
- Go to the details for the progress bar and under the slot horizontal tab change the size to fill.



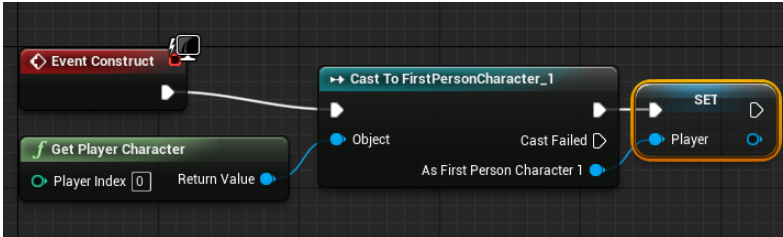
- This is what the final should look like.



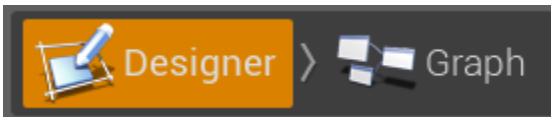
- Now we need to edit the blueprint go to the graph by selecting the button that says graph at the top left of the screen.
- Under the event construct node right click and in the menu type in “Get Player Character” and create the node.
- Drag off of the Get Player Character node and type in “Cast to FirstPersonCharacter\_1” and create the node.
- Now right click on the As First Person Character pin on the Cast to FirstPersonCharacter\_1 and in the menu that pops up select promote to variable name it “player”.



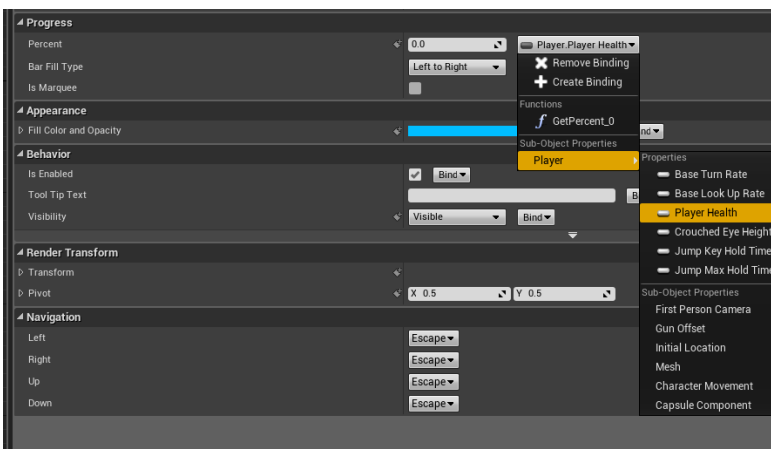
- Connect the Event Construction node with the Cast To the FirstPersonCharacter\_1 node. This is what the network should look like.



- Now go back to the designer by clicking the designer tab at the top right of the UMG editor and click on the progress in the hierarchy panel
- Go back to the Designer editor by clicking the Designer button on the top left of the screen.



- Make sure the progress bar is selected and in the details panel go to progress and next to percent on the drop down menu called bind, bind the health variable with the bar by going to the player option then in the next menu selecting Player Health



- Now change the color of the health bar by clicking on the color bar under the appearance tab. Select whatever color you want, but red is recommended
- Hit compile and then go to the content browser and drag a lava blueprint into the scene then hit play and then jump on it and the health bar should go down

## APPLICATION TIME

- Have the player's health deplete much faster when they are on the lava.

## Challenge

- Have a message display that says "Don't stand on lava!" when the player is on the lava.

# Coin Pick Up System

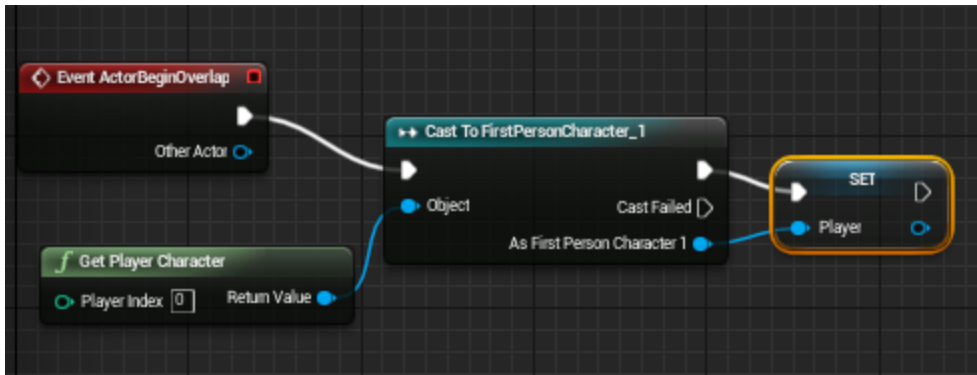
## 1. Setup

- We will create a coin pick up system for use later in the tutorial
- Create a new blueprint, make it an actor and name it "Power\_Up\_Coin".
- Open the "Power\_Up\_Coin" blueprint.
- Add a box collision component and name it trigger box.

- Set the location to 30 on the Z-axis in the details panel under the Transform category.
- Navigate to content/FirstPersonBp/Blueprint\_Tutorial\_Material/Coin and drag and drop the coin static mesh into the viewport of the blueprint

## 2. Creating the Pick Up Item

- Go to the event graph editor and right click on the grid in the menu that pops up type in “get player character” and create the node.
- On the Get Player Character Node Drag off of the Return Value pin and let go, in the menu that pops up type in “cast to firstpersoncharacter\_1” and create the node.
- Connect the Event Actor Begin Overlap node to the cast to firstpersoncharacter\_1 node.
- Now on the cast to FirstPersonCharacter\_1 node go to the “as first person character” pin which is blue right click on it and right click promote to variable, it should create a set node, name it “Player” it should connect automatically connect the as cast to first person character and the Set node.
- Here is what you should have.

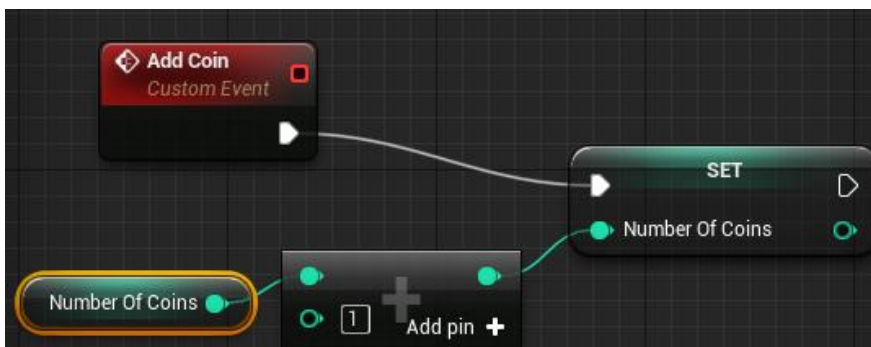


- Now that this blueprint knows how to contact the player character we need to make a couple of changes to the first person player character blueprint

### 3. Receiving the data from the picked up item

- Switch to the FirstPersonCharacter\_1 blueprint
- Find an empty place on the grid and create a new custom event by right clicking on the grid and typing in “add custom event” name it “Add Coin”.
- Now go to the green add new drop down menu and click it, in the menu that drops down add a variable named “numberOfCoins”.
- We need to make the variable an integer so with the variable selected go to the details panel and in variable type click on the drop down menu and change it to an integer.
- Now drag the “numberOfCoins” variable onto the grid and let go in the menu that pops up select get.
- Drag off of the Get numberOfCoins pin and let go type in add and select the integer+ integer option and create the node

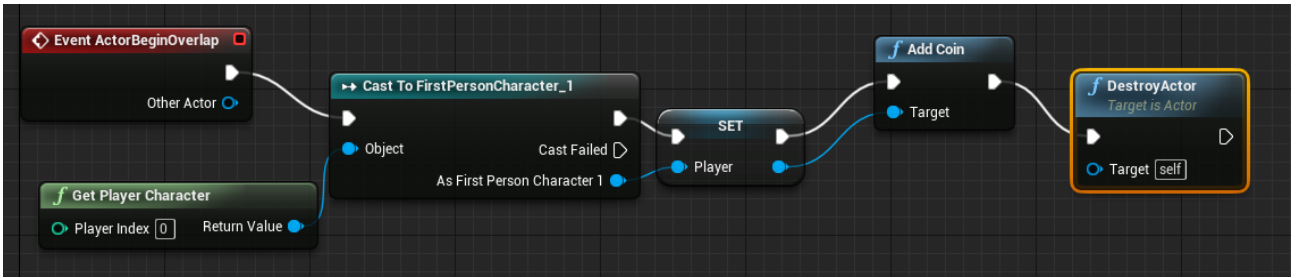
- In the Add node that was just created type in 1 as the value for the pin that doesn't have a variable attached to it, if it isn't already
- Now on the right pin of the Add node drag off of the pin and let go and type in "set number of coins" and create the node.
- Now connect the add coin event and the set amount of coins node. Here is what you should have.



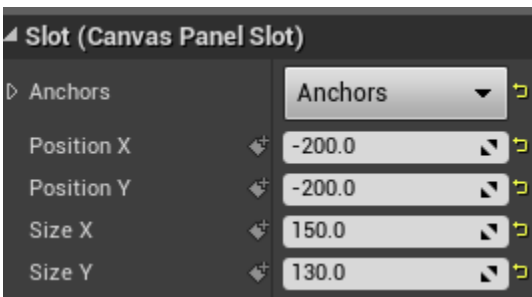
- Click on the compile button

## 4. Finishing touches for the coin Blueprint

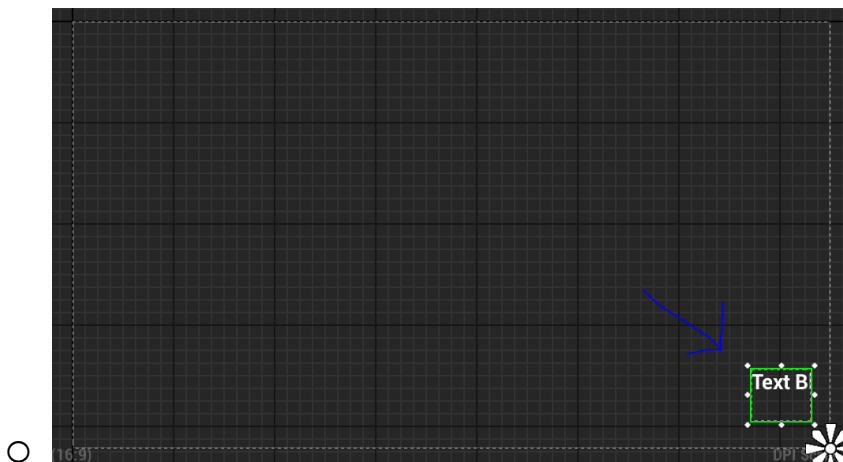
- Switch back to the Power\_Up\_coin blueprint so we can add the finishing touches.
- Now drag off of the blue pin on the left side of set player node and let go, in the menu that pops up type in "add coin" and create the node
- Now we need to get rid of the coin once the player has collected it drag of the right pin on the Add Coin event and let go type in "destroy actor" and create that node.
- Now you are done the final blueprint should now look like this.



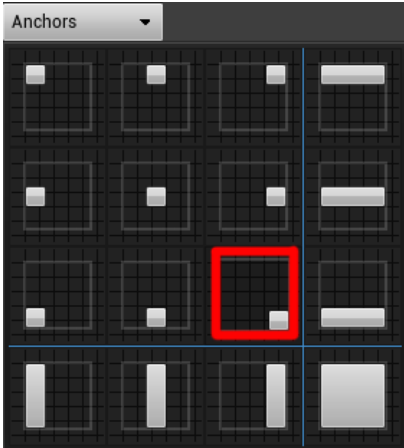
- Now we need to link the amount of coins to the HUD, switch over to the HUD widget editor and in the palette panel under panel tab drag a Horizontal Box onto the Canvas Panel
- Go to the details panel and under the Slot tab type in “-200” for the position x, “-200” for the position y, “150” for the size X, and “130” for the size y.



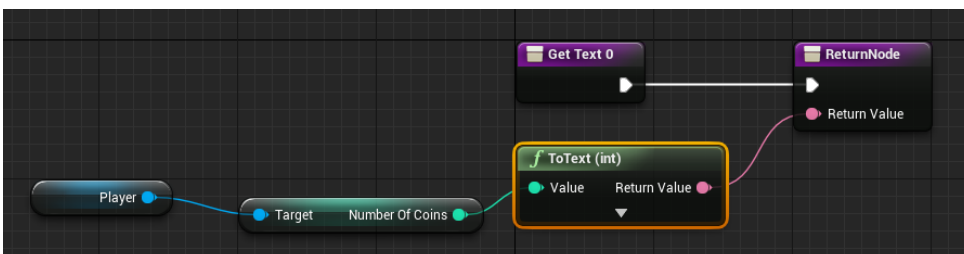
- - The Position numbers are based on 1080p monitors.
  - All you need to make sure is your “horizontal box” is positioned in the bottom right hand corner of the layout.



- Now set the anchor to the bottom right of the screen.



- Now drag a “text” item onto the new horizontal box to make it its child now click on the text box.
- Under the slot tab change the size to fill.
- Go to appearance and in the font size entry box type “40”.
- Now go to content tab and click on the bind button a menu will drop down and select “create binding” it should now take you to a grid editor.
- Drag the Player variable ,from the variables tab in the My Blueprint tab.
- Drag off of the player variable pin and type in “number of coins” and create the node.
- Now connect the Get Amount Of Coins over to the pink pin on return node unreal will say it is creating an intermediate node and when you connect it will automatically create a node between them connecting them, now hit compile.



- Drag some coins into the scene and pick them up the number at the bottom should be updating

## APPLICATION TIME

- Create a larger coin Blueprint that adds 5 to the player coin count not just 1
- OR
- Change this blueprint so that it adds 5 instead of 1.

## Challenge

- Have the coin's spinning as they await the player.

# Power Up system

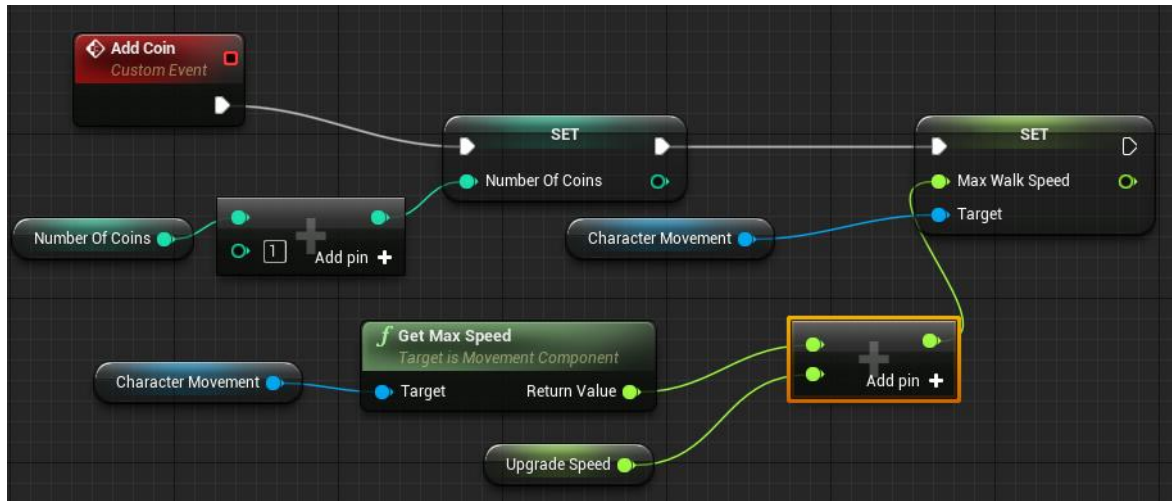
- To create a power up system we have done most of the work this one will be based on the amount of coins the player collects and the speed they will move at

## 1. Modifying the Character for power ups

- Open up the FirstPersonCharacter\_1 Blueprint if it isn't already.
- Go to the add coin event in the first person character
- Right click on grid near the Add Coin event and type in "Get Character Movement" and create the node.



- Drag off Get Character Movement of the pin let go and type in “get max speed” and create that node now near this right click on the grid and type in get max speed and select it will look like this.
- Now drag off of the return value of the “get max walk speed” node and type in add, make sure it is float + float.
- Go to the add new drop down menu and Create a new variable, name the variable “upgrade speed” then in details change the variable type to float.
- Now hit compile and set the default value to “100” under Default Value in the details panel.
- Now drag the variable onto the grid and select get from the menu that pops up.
- Connect the get upgrade speed to the open addition pin.
- Now click on the get character movement pin and hit “CTRL” and “W” at the same time to duplicate it move it to the right a little bit.
- Drag off of the new Get Character Movement pin and when you let go in the menu that pops up type in “set max walk speed” and create the node.
- Now connect the addition node to the max walk speed pin on the set max walk speed node.
- Now connect the Set Amount of Coin node to Set Max Walk Speed node. This is what you should have



- Hit **compile**, and then hit play go collect some coins and you will notice the character is moving faster each time you collect one

## APPLICATION TIME

- Have the coin change how high you can jump.

## Challenge

- Only change the player's max speed when the player gets 5 or more coins.