**Setting Up PlayerStatsBP**

* In ThirdPersonBP folder, right-click on empty space, and click Blueprint Class
  + Select parent class Actor, name it PlayerStatsBP
* Double-click PlayerStatsBP to enter the Blueprint editor
  + Add the following components: Cube, Box Collision, Particles (optional), Rotating Movement (optional)
  + Click and drag Cube into Default Scene Component (to make the Cube the Default Component)
  + If it has not been done so already, click and drag the box collision and particles into Cube to attach. Rotating movement will **not** attach to scene component.
* Add the Following Variables: CurrentHP, MaxHP, IncreaseHP, CurrentMP, MaxMP, IncreaseMP, CurrentXP, MaxXP, IncreaseXP
  + Under Details, Add variables to categories for better organization. Attrs for stats, Bools for Boolean variables
* Also add bIsPowerup, bIncreaseCurrentHP, bIncreaseMaxHP, bIncreaseMaxMP, bIncreaseCurrentMP, bIncreaseCurrentXP, bIncreaseMaxXP, bHasParticleEffect, bIsRotating
  + The ‘b’ in front of each variable name helps denote a bool, also prevents variable and function naming issues later on.
* Add reference variables: ThirdPersonCharacter, PlayerStatsBP, and EStatesPowerup
  + To add as reference, click on Variable Type dropdown menu, and type in PlayerStatsBP. Highlight PlayerStatsBP and click Object Reference. Make sure EStatesPowerup and PlayerStatsBP are public.

**Setting up EStatesPowerup**

* In ThirdPersonBP, right-click and highlight Blueprints, the click Enumeration. Name it EStatesPowerup
* Click ‘New’ and add the following Enums: EPowerup, ECurrentHP, EMaxHP, ECurrentXP, EMaxXP, ECurrentMP, EMaxMP.
* Click Save
* You were not able to add EStatesPowerup as a reference earlier, but you can do so now. This will allow the use of a switch statement later on.

**Setting up M\_PowerupParticles**

* In ThirdPersonBP, make a new folder, called Materials.
* Right-Click and click Material and name it M\_PowerupParticles
* Double-Click the new material to enter the editor
* Click the main M\_PowerupParticles node and select the Translucent Blend Mode in the Details pane. This will unlock the Opacity output.
* Right-click in the Blueprint area and type in RadialGradientExponential and hit enter.
* Right-click again and type in Particle Color and hit enter
* Hold down the ‘M’ key and left-click on the Blueprint area to bring up the Multiply node. You will need two of these.
* Drag off from RadialGradientExponential and connect to both A inputs in the Multiply nodes
* Drag off from the Top and Bottom (white) Particle Color inputs and connect to both B outputs in the Multiply nodes.
* Click save and you are done with the material.

**Setting up PowerupParticles**

* In ThirdPersonBP, right-click, then click Particle System, name it PowerupParticles.
* Double-click the new Particle system to enter the editor.
* Click on Required
  + Set the material for the Particles (M\_PowerupParticles).
  + All other setting remain at default
* Click on Spawn
  + Expand Rate and Distribution
    - Set value to 5
* Click on Lifetime
  + Set Distribution to Distribution Float Constant
  + Constant: 1.25
* Click on Initial Velocity
  + Max: 30, 30, 30
  + Min: 0, 0, 0
* Click on Color Over Life
  + Expand Color Over Life
  + Set Distribution to Distribution Vector Constant Curve
  + Expand Constant Curve and Points
    - Change OutVal to white: RGB: 1, 1, 1 for both Points 0 and 1
  + Expand Alpha Over Life
  + Set Distribution to Distribution Float Constant Curve
  + Expand Constant Curve and Points
    - Bring Array element up to 5
    - 0: InVal: 0.0 , OutVal: 1.0
    - 1: InVal: 0.25 , OutVal: 1.0
    - 2: InVal: 0.0 , OutVal: 0.75
    - 3: InVal: 0.0 , OutVal: 1.0
    - 4: InVal: 1.0 , OutVal: 0.0
* Add two more now fields: Location -> Sphere, Location -> Initial Location
  + Leave all values at Default.
  + You can add other effects like Rotation and Orbit if you like. Have fun with it!
  + Click save and you are done with the Particles.

**Other Settings in PlayerStatsBP**

* Rotating Movement
  + The only thing you need to adjust here is the rotation rate under detail (Z Axis). I have it set to 100.
* Now that particles and rotation are all set, we need to be able to control when they are activated
  + In details, type in Activation and check off the box that says Auto Activate. This will prevent AutoActivate from overriding code, and we can set it to activate with each instance.
* In the PlayerStatsBP editor, right-click on an open grid area, and type in Event Begin Play
  + Drag off from here and create a Branch node. Alternatively, you can hold B and left-click.
  + Drag out the HasParticleEffect bool and connect it to Condition on the Branch node.
  + From the True input, drag out and Type in Activate and select Activate under Components
  + From False, drag out and type Deactivate, and Select Deactivate under Components
  + Last thing is to connect the Particle system.
    - From Component (top-left) simply drag out Particles and connect to Target on Activate. You can also connect the same Particle node to Target on Deactivate
* Follow the same steps to set up IsRotating with the Rotating Movement component, except you will have to use Event Tick instead.
  + With this setup, you can now edit each instance of PlayerStatsBP to have particles and/or rotation (edit under Defaults).

**Setting up PlayerStatsBP scripts**

* Now we finally start setting up the functions for the PlayerStatsBP
* Right-click on Box under components, highlight Add Event, then click Add On Component Begin Overlap
* From the White input, drag off and type CastToThirdPersonCharacter and hit enter
  + Connect Object from the cast node to Other Actor to OnComponentBeginOverlap
* From the Right side White input of the Cast node drag off and type SwitchOnEstatesPowerup and hit enter. This brings up the switch statement which will direct the execution flow to the appropriate function. You don’t need to connect EPowerup to anything.
* From MyBlueprint, drag out EStatesPowerup and connect it to the Selection output on the Switch node.
* Create a new function called IncreaseCurrentHP.
  + Drag out from ECurrentHP and type IncreaseCurrentHP to select the function we just created.
  + Double-click on the function to open it. First, click on the Purple function node to edit the inputs. Add an input called OtherActor. This will connect to another cast node for ThirdPersonCharacter.
    - Return to the Event graph. You will see that OtherActor has been added to the function node. Connect this to the OtherActor input on the collision event. (You may want to start using reroute nodes).
  + Back in IncreaseCurrentHealth, connect the purple node to a CastToThirdPersonCharacter node. Connect Object to Other Actor
  + Connect the Cast node to a Branch
    - Drag out the IncreaseCurrentHP bool and connect it to Condition on the branch node.
    - Drag out from As ThirdPersonCharacter and type Get CurrentHP. You will notice that the new node will disconnect. What you need is a Target input as well. In order to do this, you need to add these variables to ThirdPersonCharacter as well, since this is being cast to that. You will only need the Attrs variables, not the bools.