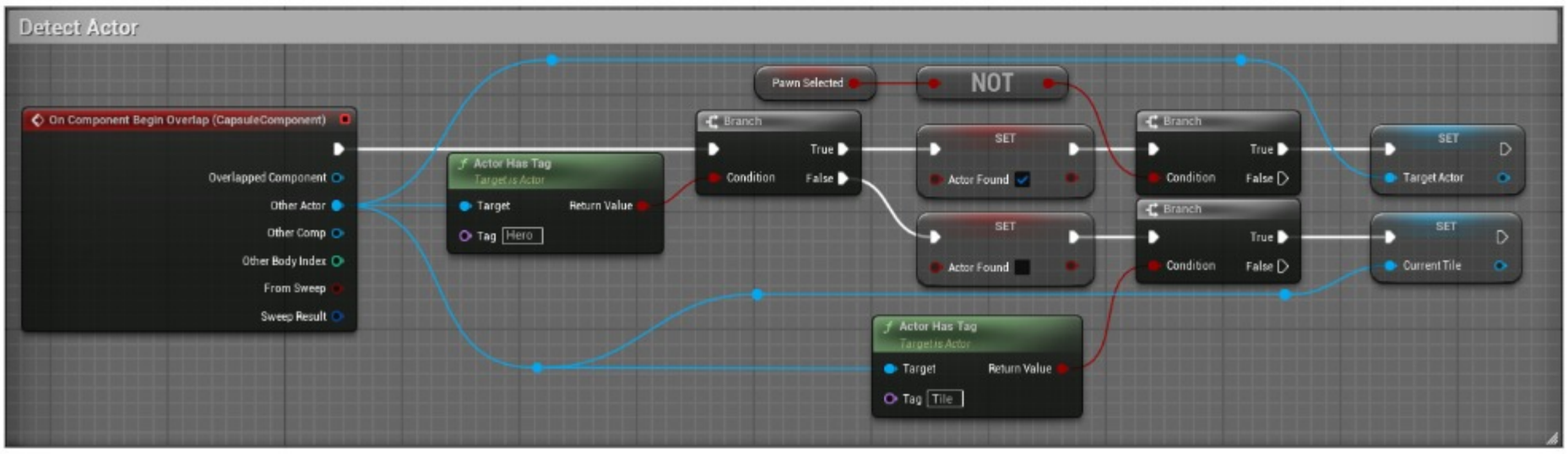
Tile Selector - Actor
Feature - Detecting, Picking Up, and Manipulating Other Actors

BP_TileSelector

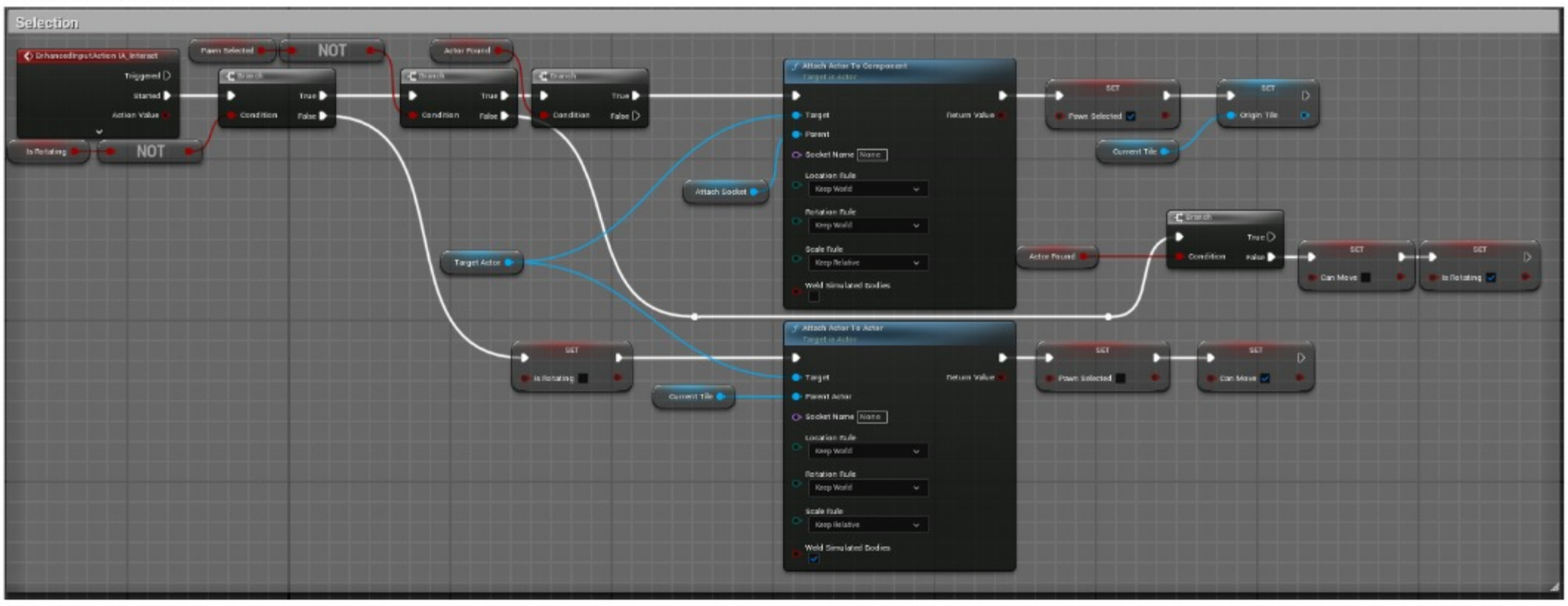
Involved Variables
PawnSelected - Bool
ActorFound - Bool
TargetActor - Actor reference
CurrentTile - Actor reference
OriginTile - Actor reference
AttachSocket - Scene Component Object reference
IsRotating - Bool
CanMove - Bool
CurrentRotation - Rotator

Involved Functions
Detect Actor - Event 'On Component Begin Overlap (CapsuleComponent)'
Selection - Event 'EnhancedInputAction IA_Interact
Return - Event 'EnhancedInputAction IA_Return
Movement - Event EnhancedInputAction IA_Move
Save Current Rotation - Custom Event

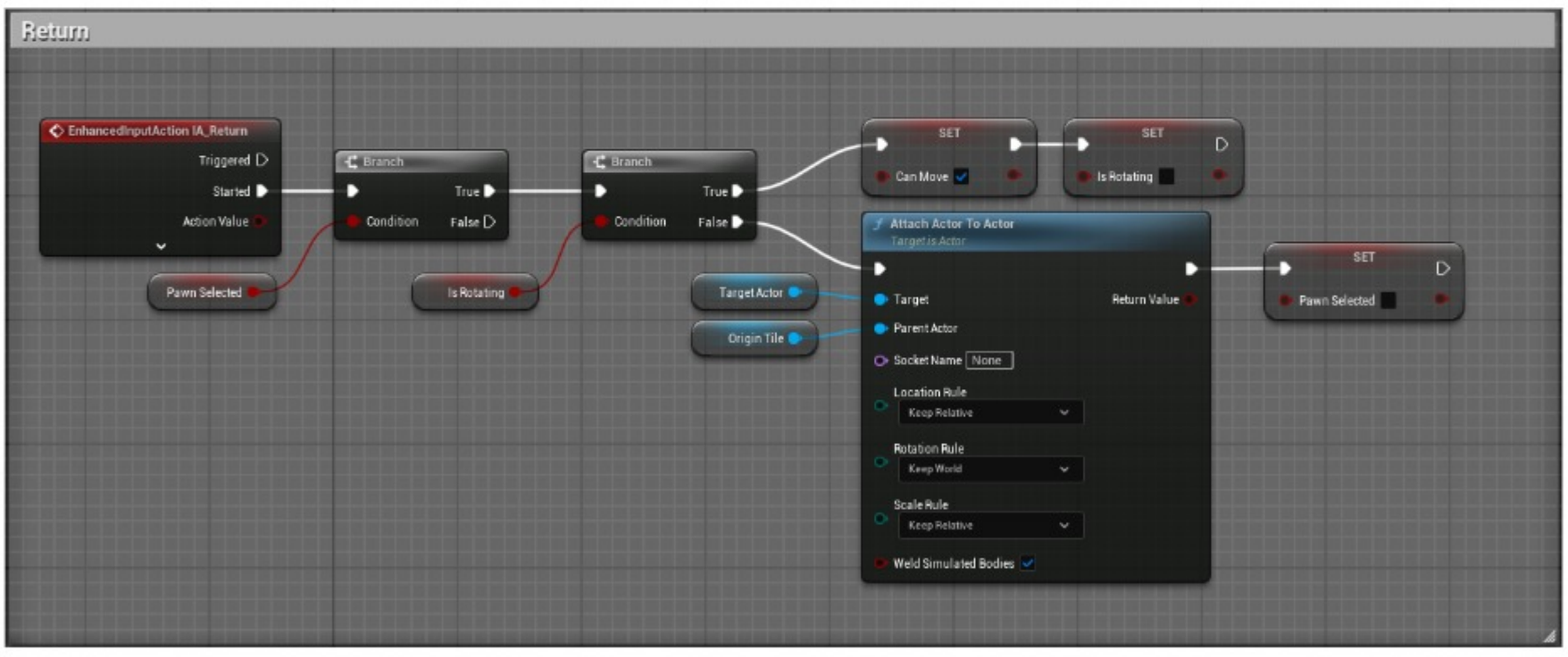
Detect Actor Function



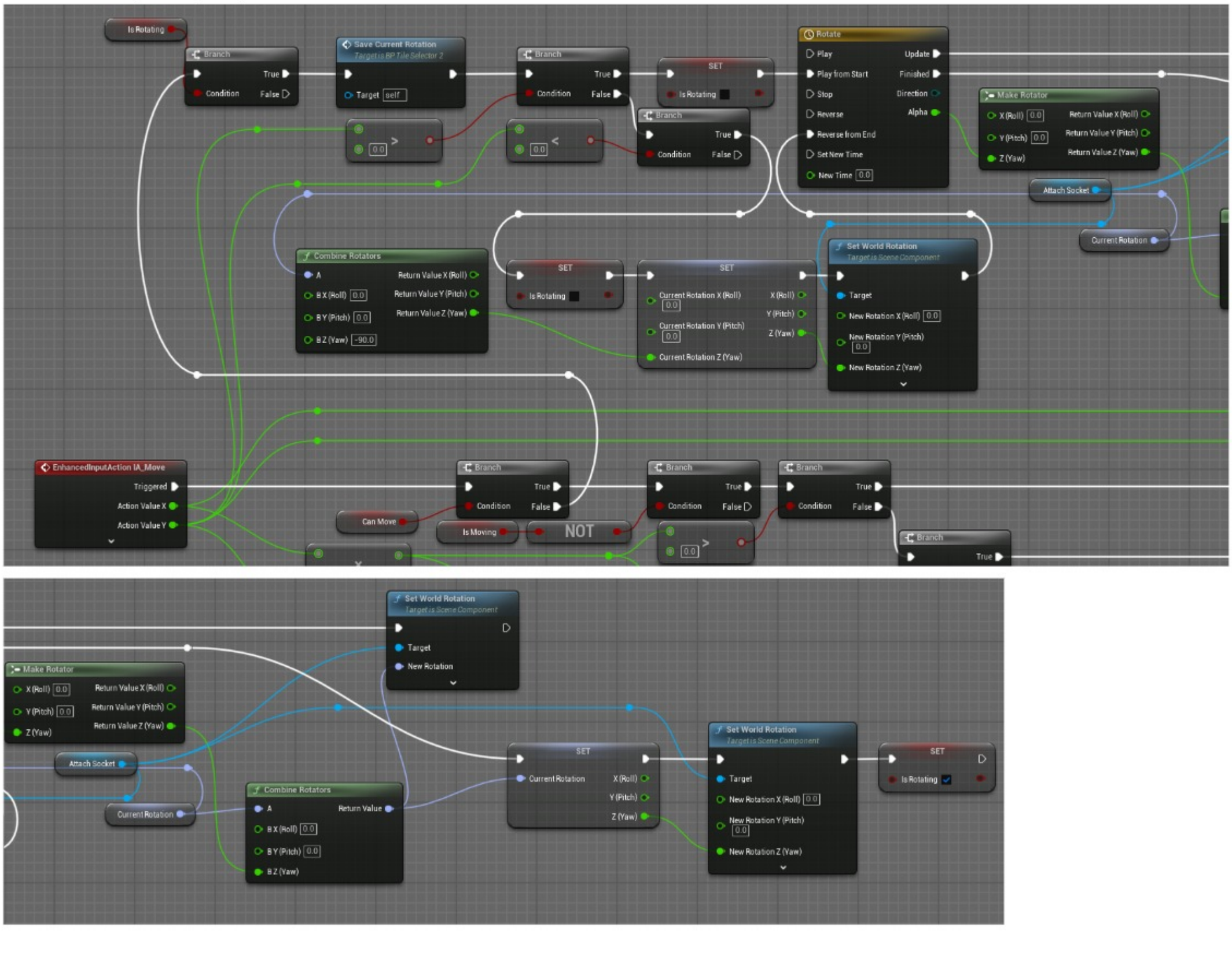
Selection Function



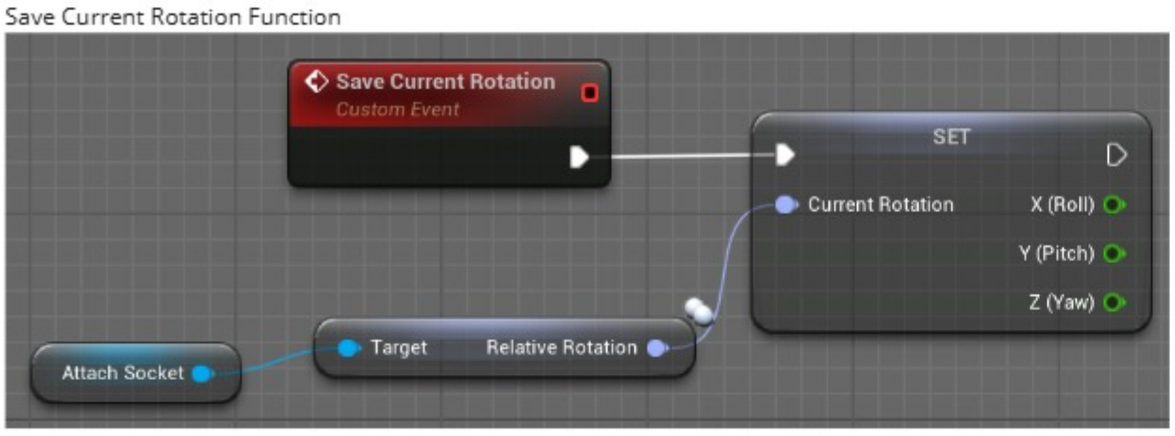
Return Function



Movement Function (Rotation portion)



Save Current Rotation Function



Pseudo Code

Detect Actor

When the Capsule Component of the Tile Selector Actor begins overlapping
- If Other Actor has Tag 'Hero'
  - Set ActorFound bool to True
  - If bool PawnSelected is NOT True
    - Set TargetActor object reference to Other Actor
- Else set ActorFound bool to False
- If Other Actor has Tag 'Tile'
  - Set CurrentTile object reference to Other Actor.

Selection

When Input Action IA_Interact is Started
- If bool IsRotating is NOT True
  - If bool PawnSelected is NOT True
    - If bool ActorFound is True
      - Attach actor TargetActor to component AttachSocket, keeping world location and rotation.
      - Set bool PawnSelected to True
      - Set OriginTile object reference to CurrentTile
  - If bool PawnSelected is True
    - If bool ActorFound is False
      - Set bool CanMove to False
      - Set bool IsRotating to True
- If bool IsRotating is True
  - Set bool IsRotating to False
  - Attach actor TargetActor to actor CurrentTile, keeping world location and rotation.
  - Set bool PawnSelected to False
  - Set bool CanMove to True

Return

When Input Action IA_Return is Started
- If bool PawnSelected is True
  - If bool IsRotating is True
    - Set bool CanMove to True
    - Set bool IsRotating to False
  - If bool IsRotating is False
    - Attach actor TargetActor to actor OriginTile, keeping relative location and world rotation.
    - Set bool PawnSelected to False.

Movement

When Input Action IA_Move is Triggered
- If bool CanMove is False
  - If bool IsRotating is True
    - Call custom Event Save Current Rotation
    - If IA_Move Action Value Y is greater than 0 is True
      - Set bool IsRotating to False
      - Play Timeline 'Rotate' from start with duration 0.2 seconds and float value Alpha of 90.
        - On Update, set World Rotation of component Attach Socket using Rotator combining CurrentRotation Rotator and the Z (Yaw) value determined by Timeline's Alpha value.
        - On Finished, set CurrentRotation Rotator to Rotator combining CurrentRotation Rotator and the Z (Yaw) value determined by Timeline's Alpha value.
      - Set World Rotation Z(Yaw) value of Attach Socket Component to CurrentRotation Z value.
      - Set bool IsRotating to True.
    - If IA_Move Action Value Y is greater than 0 is False
      - Set bool IsRotating to False
      - Set Rotator CurrentRotation to CurrentRotation with Z value -90
      - Set World Rotation of Attach Socket Component to CurrentRotation
      - Play Timeline 'Rotate' reversed from the end.
        - On Update, set World Rotation of component Attach Socket using Rotator combining CurrentRotation Rotator and the Z (Yaw) value determined by Timeline's Alpha value.
        - On Finished, set CurrentRotation Rotator to Rotator combining CurrentRotation Rotator and the Z (Yaw) value determined by Timeline's Alpha value.
      - Set World Rotation Z(Yaw) value of Attach Socket Component to CurrentRotation Z value.
      - Set bool IsRotating to True.

Save Current Rotation

When Event is called
- Set CurrentRotation to Relative Rotation of Attach Socket Component.

Summary and Explanation

Detect Actor

In this function, we have the Capsule Component of the Tile Selector actor detecting overlapping actors. Important actors are manually marked with tags to differentiate them, marking them as 'Hero' actors or 'Tile' actors. We first check for the 'Hero' tag to detect if our selector has now overlapped with a 'Hero' actor that we can interact with. If it has, we make a note with the ActorFound bool and then check the PawnSelected bool to make sure the Tile Selector is not already carrying another actor. As long as it is not carrying another actor, the current overlapped 'Hero' actor is set as the TargetActor reference, marking it as a possible actor to interact with.

If the overlapped actor is not a 'Hero', the function says there is no actor found through the ActorFound bool and then checks if it is instead a 'Tile' actor. If it is, we set a reference to it with the CurrentTile object reference.

The objective of this function is primarily to set these two object references, TargetActor and CurrentTile, for use in the next two functions.

Selection

The purpose of this function is to make use of the object references set in the previous function and to 'pick up' the TargetActor or to then 'drop' it on the CurrentTile. This function starts when it first detects the Input Action key for 'Interact', which I set to the F key. In order to accommodate a rotation mechanic to manipulate the orientation of the picked up Actor, the function first checks for the value of the IsRotating bool. If it is found to be 'false', a second check is made to confirm if PawnSelected is 'true', i.e. if the Selector is currently carrying an actor. If this is found to be 'false', meaning no actor is currently being carried, the function then checks if ActorFound is true, meaning there is an actor to interact with overlapping the cursor. If it is, the function then attaches the TargetActor set in the Detect Actor function to the AttachSocket component of the Tile Selector actor, setting it as a child component that will now move with the Tile Selector actor. We keep the world location and rotation to ensure the target actor does not snap to the Tile Selector and maintains its current location and orientation. The PawnSelected bool is then set to 'true' to announce the Tile Selector is now holding an actor, and the OriginTile object reference is set to the same as the CurrentTile reference set in the Detect Actor function in order to store a note of where this actor was picked up from for use later.

If PawnSelected is found to be 'true' in the earlier check, meaning the Tile Selector is already carrying an actor, we instead want to be able to place the actor rather than pick a new one up. The function therefore checks if the ActorFound bool is 'true', meaning if we are currently overlapping another actor. If we are, we cannot place the actor we are carrying on top of it, so we need ActorFound to be 'false' If it is, the function could then attach the carried actor, but in order to instead enter a rotation feature, we set the CanMove bool to 'false' and the IsRotating bool to 'true'.

With this logic, we then return to the IsRotating bool check at the start. If this bool is then found to be 'true', we set it to 'false' to end the rotating stage and attach the TargetActor we are carrying to the CurrentTile we are now hovering over. This will detach it from the Tile Selector actor, leaving it placed on the free tile. Again, we keep world location and rotation in order to prevent the TargetActor from snapping to the tile, ensuring it maintains its current position and rotation. We then set the PawnSelected bool to 'false' to announce we are no longer carrying an actor and set the CanMove bool to 'true' so we can move again.

Return

The purpose of this function is the cancel an action or return to a previous state, either returning the carried actor back to where it was picked up from, or backing out of the rotation stage to move again without placing the actor. This function is triggered only when it first detects the Input Action for return has been triggered, in my case either the right mouse button or the C key, for cancel. First it checks whether PawnSelected is 'true', meaning the Tile Selector is currently carrying an actor. It then checks if IsRotating is 'true', meaning the Tile Selector is currently stationary and rotating the carried actor. If this is 'true', it backs out by resetting the CanMove bool to 'true' and the IsRotating bool to 'false'.

If IsRotating is found to be 'false', the Tile Selector must now be carrying an actor and able to move with it. Therefore, the return key will now return the carried actor to its original location, by attaching the TargetActor to the OriginTile reference that we set when we first picked it up. We keep the world rotation, but this time we use the relative location so that the carried actor snaps back to the tile it was picked up from. Finally, we set the PawnSelected bool to 'false' to announce that the Tile Selector is no longer carrying anything and is free to pick up another actor.

Movement

The rotation mechanic is being added to the Movement Function as it will be using the same keys, specifically A and D. It has it's own branch by detecting first if bool CanMove is False, and then that IsRotating is True. Here it calls the custom Event Save Current Rotation first thing to have the Rotator as a reference. It then checks which direction it should be rotating by determining whether the Action Value received from the initiating Event is greater or less than zero.

If the value is greater than zero, we will be rotating clockwise. We directly enter a Timeline aptly named 'Rotate' from the start. This Timeline is 0.2 seconds long, determining the length of our rotation animation, and has a float value called 'Alpha' of 90. This value will go from 0 to 90 relative to the elapsed time, hitting 90 at the end of the 0.2 seconds. On Update, which means every tick that the Timeline is active, the function sets the World Rotation of the Attach Socket component where the carried Actor is currently attached after the Selection Function. The new rotation is a Rotator combining the CurrentRotation Rotator and a Z(Yaw) value determined by the Timeline's Alpha value, meaning the yaw of the Attach Socket will increase 90 degrees in 0.2 seconds. When the Timeline is Finished, we cement these changes by setting the CurrentRotation Rotator to the value of the combined Rotator at the last frame of the Timeline, where the Alpha value is 90. We then set the World Rotation of the Attach Socket to this new CurrentRotation value. Lastly, we return the bool IsRotating to True so that the function is open to new inputs, signifying the rotation is done and the Actor can either be rotated again or placed.

If the value of the initiating Event Action Value is less than zero, we enter a different branch to rotate counter clockwise. The bool IsRotating is set to False to prevent further inputs until the rotation is completed. We then set the CurrentRotation Rotator equal to the CurrentRotation with a Z(Yaw) value of -90. The function then uses this updated value to immediately change the Attach Socket's World Rotation before entering the same Timeline, but reversed from the end. The reason we set the rotation directly before we enter the Timeline is because the Timeline, when reversing from the end, will snap the Attach Socket to 90 degrees past its current rotation before gradually moving it back to 0, instead of going from 0 to 90 as we do to turn clockwise, or going from 0 to -90 as we intend. By setting the rotation to -90 and then immediately entering the Timeline, we have the Timeline go from 0 to -90 as intended. From there we join up again with the clockwise portion, following the Update and Finished paths described above.

Save Current Rotation

This short custom Event exists only to save the Attach Socket components current Relative Rotation with our CurrentRotation Rotator. It ensures our Rotator has an up-to-date value of the rotation by setting it at the start of the rotation portion of the Movement Function.