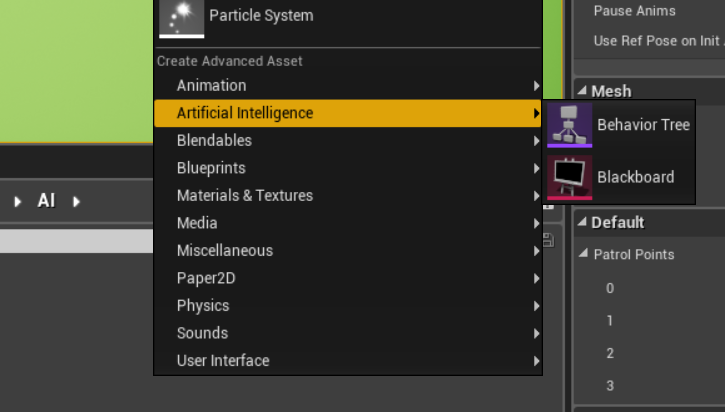
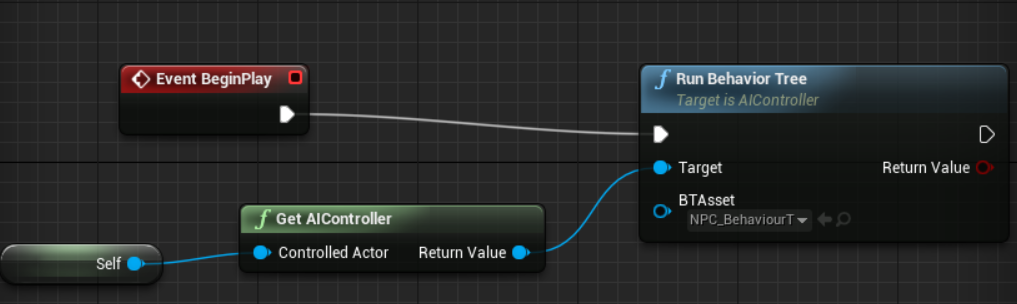
**Week 8 – AI Behaviour Trees**

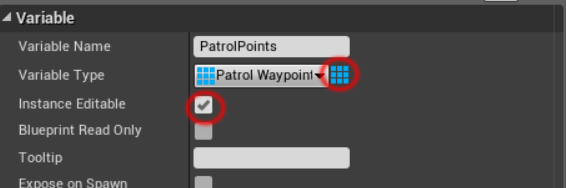
*For reference on behaviour trees please see the lecture (kindly shared by Ade) or the UE4 docs.*

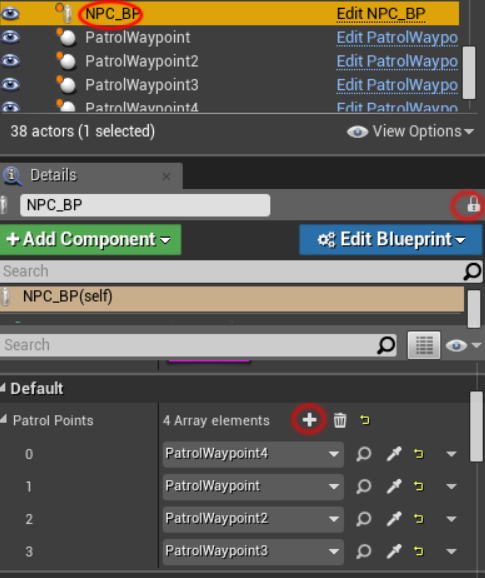
**Section 1 – Creating a new behaviour tree**

1. Create a new folder inside your blueprints folder called “AI”
2. Create a new behaviour tree called “NPC\_BehaviourTree” and a Blackboard called “NPC\_BB\_Data”
3. Open up your NPC\_BP blueprint editor and break the links for the AI scripting we did last week
4. On Begin Play add the following:



**Section 2 – Patrolling Setup**

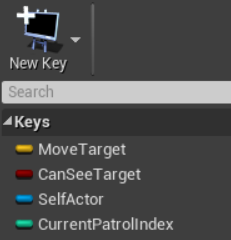
1. Create a new blueprint class deriving from Actor, call it “PatrolWaypoint”
2. Drag 4 of them into the scene
3. Add a new variable in NPC\_BP blueprint editor called PatrolPoints
4. Make sure to click on the icon to the right of the variable type and change it to the array icon
5. Also select Instance Editable.
6. Compile and save
7. Go into the scene and find your NPC that you placed before
8. Lock this details panel by selecting the lock symbol upper right.
9. Add 4 elements to the Patrol Points array and drag the patrol points into the slots.



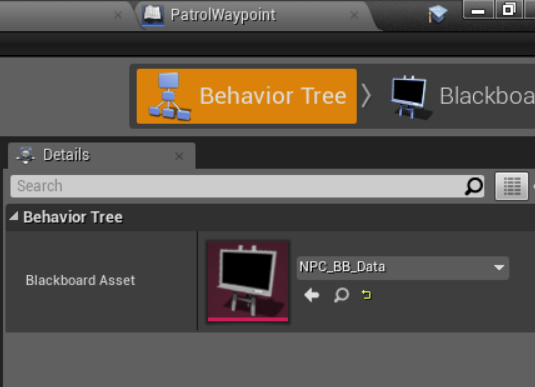
1. Save the map.

**Section 3 – Setting up the blackboard**

1. Open up your NPC\_BB\_Data blackboard you created earlier
2. Add the following:
3. Vector MoveTarget
4. Bool CanSeeTarget



1. Open up your new behaviour tree
2. On the right details panel set the Blackboard Asset to NPC\_BB\_Data and save.

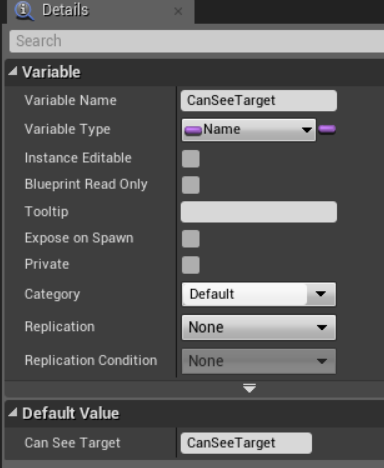


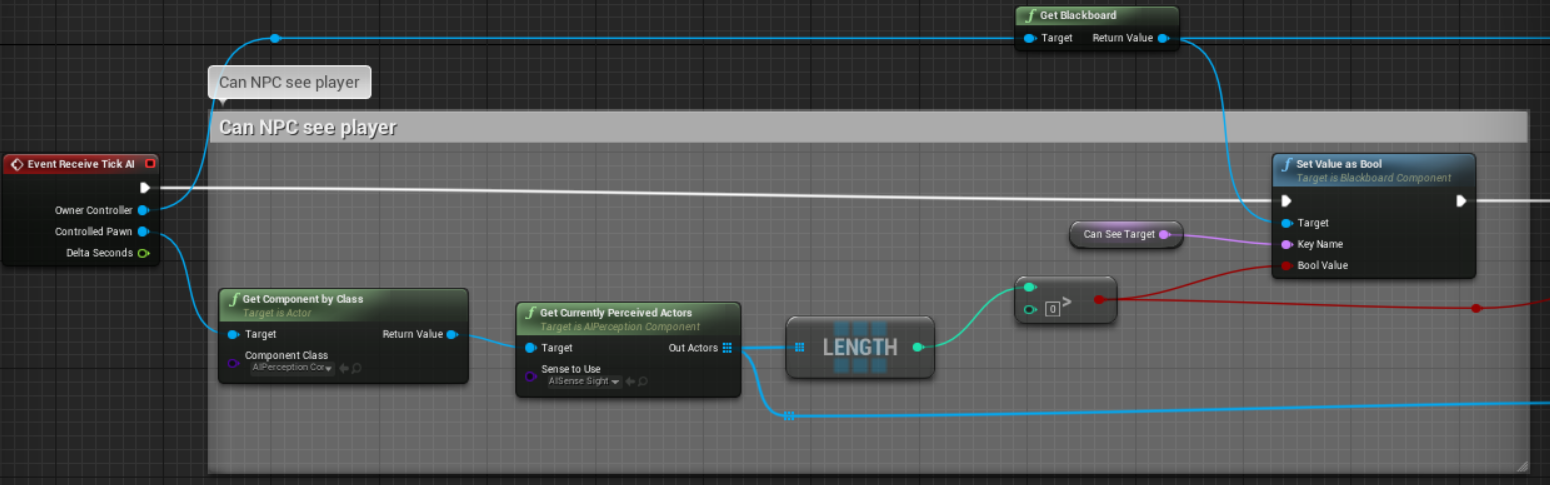
**Section 4 – Services**

*Services are processes run by the behaviour tree to update any conditions or variables.*

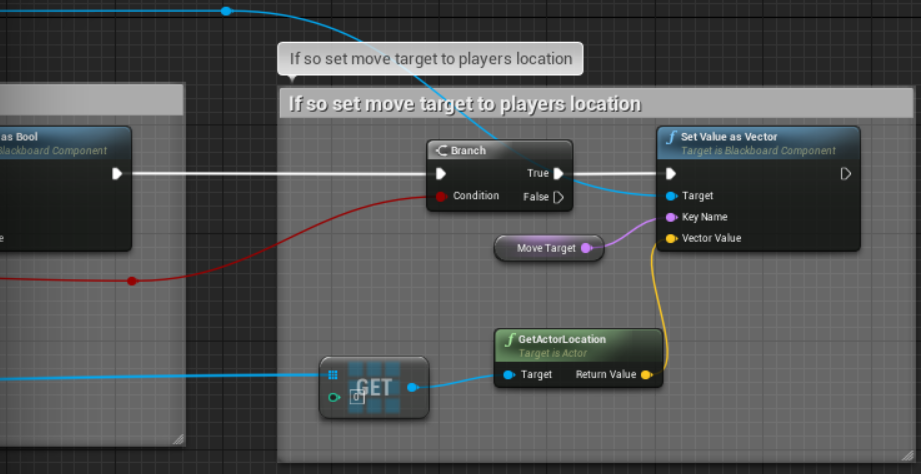
The first service we are going to create is going to use the AIPerception component to update the CanSeePlayer Boolean on the blackboard, if we can see the player we will also update the MoveTarget positional vector.

1. Click the New Service button and select derive from BTService\_BlueprintBase
2. A new blueprint editor window will open, return to the editor and rename it BTService\_LookForPlayer
3. Create a Receive Tick AI node and replicate the code below
4. When you get to the SetValue as Bool node right click on Key Name and select promote to Variable
5. It is important that you then compile and then set its default value to “CanSeeTarget” or exactly as you spelled it in the blackboard.





1. Next replicate the remainder of the code below to set the MoveTarget blackboard variable if the player has been seen.

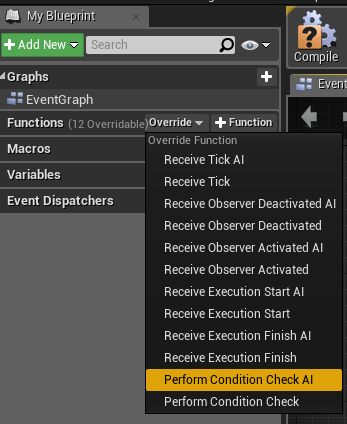


**Section 5 – Decorators**

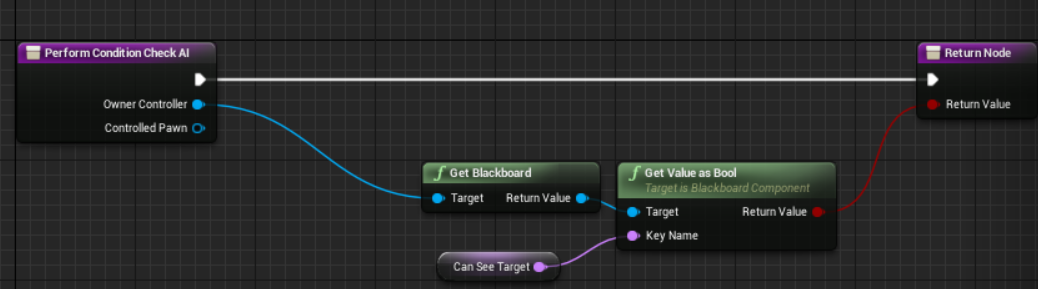
*A decorator is a run by the behaviour tree but must return a true or false. They are mainly used in conjunction with selector composite nodes.*

We are going to create a simple decorator that simply returns a value from the blackboard, in this case whether or not the NPC can see the player.

1. Create a new Decorator the same way we created the Service but call it BTDecorator\_CanSeeTarget
2. Override the function called Condition Check AI by selecting the Override drop down next to Functions on the left panel.



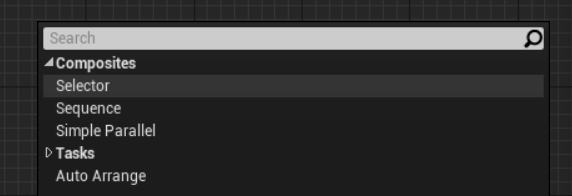
1. Recreate the code below:



1. Remember to create the Key Name by right clicking on it and selecting Promote to Variable then compiling and setting the default value to “CanSeeTarget”.

**Section 6 – The Behaviour Tree Graph**

1. Right click on empty space in the graph and select Composites->Selector

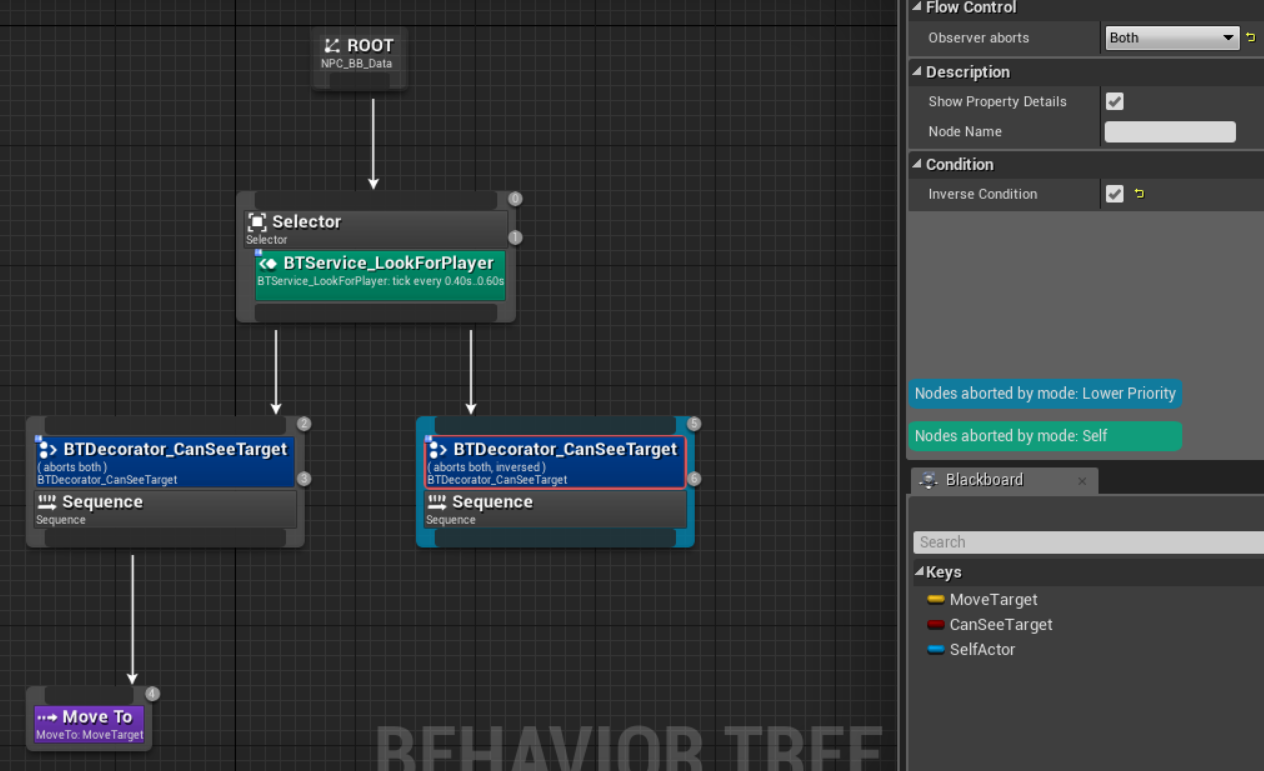


1. Make sure the root is connected to the selector
2. Right click on the selector then select Add Service->BTService Look For Player



1. Add two sequence composite nodes and attach BTDecorator\_CanSeeTarget to them both
2. On the right side one select Inverse Condition true
3. On both of them set Observer aborts to Both

*This means if the condition changes while the sequence is executing it will return up the tree.*



1. Right click and create a MoveTo node and set the Blackboard Key to MoveTarget.

*We have now recreated the functionality we set up last week but in a much more robust way, as long as the pawn has an AIPerception component any pawn we create can use this behaviour tree.*

Section 7 – Tasks

*A task is always an end point of a branch of the tree, it is used to actually “do” things with the pawn.*

In the demo I went over how to create a patrolling pawn, I am not going to give this to you step by step as it is a good practise exercise.

To get you started here are some steps to follow:

1. Use Receive Tick AI node as a starting point
2. Create variables inside the Tasks editor for holding the patrol points array and tracking the current patrol point index
3. Use a Sequence node to split up the execution of tasks
4. Task 1 – If patrol points array is empty, get the array from the pawn (cast to NPC\_BP)
5. Task 2 – Check if the pawn is near to its target patrol point, if so increase the index (use modulo)
6. Task 3 – Move to the patrol point at the current index in the array.
7. Add the task to the behaviour tree if the CanSeeTarget decorator is false.

Challenges

1. Health Awareness - Add a service that checks the NPC’s health and sets a blackboard bool if it is low. If that bool is true then they find the nearest health pick up and move to it.
2. Self Destruction - Add a service that checks if the player is within close range, if so create a task that calls a custom event in the NPC’s blueprint that causes them to explode, damaging anything nearby.